Sun Java Enterprise System 5 インストール計画ガイド

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820–0879
2007 年 2 月

Copyright 2007 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 U.S.A. All rights reserved.

本書で説明する製品で使用されている技術に関連した知的所有権は、Sun Microsystems, Inc. に帰属します。特に、制限を受けることなく、この知的所有権には、米国特許、および米国をはじめとする他の国々で申請中の特許が含まれています。

U.S. Government Rights – Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

本製品には、サードパーティーが開発した技術が含まれている場合があります。

本製品の一部は Berkeley BSD システムより派生したもので、カリフォルニア大学よりライセンスを受けています。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびにほかの国における登録商標です。

Sun、Sun Microsystems、Sun のロゴマーク、Solaris のロゴマーク、Java Coffee Cup のロゴマーク、docs.sun.com、Java、 Solaris は、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。Sun のロゴマークおよび Solaris は、米国 Sun Microsystems 社の登録商標です。すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用している同社の米国およびその他の国における商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャーに基づくものです。

OPEN LOOK および Sun™ Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザーおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカルユーザーインタフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは、OPEN LOOK GUI を実装するか、または米国 Sun Microsystems 社の書面によるライセンス契約に従う米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

この製品は、米国の輸出規制に関する法規の適用および管理下にあり、また、米国以外の国の輸出および輸入規制に関する法規の制限を受ける場合があります。核、ミサイル、生物化学兵器もしくは原子力船に関連した使用またはかかる使用者への提供は、直接的にも間接的にも、禁止されています。このソフトウェアを、米国の輸出禁止国へ輸出または再輸出すること、および米国輸出制限対象リスト (輸出が禁止されている個人リスト、特別に指定された国籍者リストを含む) に指定された、法人、または団体に輸出または再輸出することは一切禁止されています。

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われないものとします。

070830@18135

目次

はじめに 5

1 インストール計画の概要 11
このリリースで使用される Java ES コンポーネント 11
Java ES インストールの定義 12
インストール計画のタスク 13

2 実装の仕様の作成 15
配備アーキテクチャーの分析 15
コンピュータハードウェアおよびオペレーティングシステム仕様の作成 17
ネットワーク接続仕様の作成 18
ユーザー管理仕様の作成 20
ソリューションの LDAP スキーマの指定 20
ソリューションのディレクトリツリー構造の指定 21

3 インストール計画の準備 25
インストール計画の問題 25
分散インストール 26
コンポーネントの依存関係 26
相互動作のための設定 32
冗長性戦略 33
LDAP スキーマと LDAP ディレクトリツリーの構造 34
Java ES インストーラの動作 35
インストール計画の作成 40

A Java ES と Solaris 10 ゾーン 45
ゾーンとは何か 45

複数ゾーン環境の構造 46
完全ルートゾーンと疎ルートゾーンのどちらを選択するか 47
パッケージの伝播 47
Java ES にゾーンを使用する理由 48
Java ES コンポーネントのゾーン制限 49
Java ES 共有コンポーネントとゾーン 49
Java ES 製品コンポーネントとゾーン 51
Java ES インストーラでのゾーンサポート 52
Java ES 伝播ポリシー 52
製品コンポーネントのインストール 53
製品コンポーネントのアップグレード 53
すべての共有コンポーネント 55
共有コンポーネントに関連する Java ES インストーラ動作の要約 55
Java ES で推奨されるゾーンの使用法 57
推奨されている実践事項 59
配備アーキテクチャー 61
特殊なケースまたは例外 61
製品コンポーネントの特殊ケース 61
共有コンポーネントの特殊ケース 62
サンプル: 疎ルートゾーンに Application Server をインストールする 62

索引 65

はじめに

Sun Java™ Enterprise System (Java ES) ソリューションのインストールは広範なプロセスです。一般的なソリューションでは、ネットワークに接続された複数のコンピュータに Java ES コンポーネントをインストールし、相互動作するコンポーネントインスタンスを構成します。本書『Sun Java Enterprise System インストール計画ガイド』では、Java ES のアーキテクチャーを分析する方法と、ソリューションをインストールするための計画を作成する方法について説明します。

対象読者

インストール計画は、Java ES ソリューションライフサイクルの 1 つの段階です。『インストール計画ガイド』では、『Sun Java Enterprise System Deployment Planning Guide』で説明されているライフサイクルの初期段階が完了しており、配備アーキテクチャーとして知られる、ソリューションの上位レベルの技術的な説明書が作成済みであることを前提とします。

『インストール計画ガイド』は、配備アーキテクチャーを分析し、ソリューションをどのようにインストールおよび設定するかを決定する方を対象としています。

このマニュアルをお読みになる前に

『インストール計画ガイド』では、ソリューションライフサイクルの全段階を 1 人の人物が実行することを想定していません。インストール計画は、次の知識を持つ担当者が作成することをお勧めします。

- Java Enterprise System を構成するコンポーネントと、各コンポーネントによって提供されるサービスについての一般的知識。詳細については、『Sun Java Enterprise System 5 技術の概要』の「Java ES コンポーネント」を参照してください。
- ネットワークアドレスの設定、負荷分散用のハードウェアまたはソフトウェアの使用方法、ファイアウォールを利用したネットワークのセキュリティー強化、DNS サーバーの設定などについての知識を含む、IP ネットワーク設定についての十分な理解。

- これには、オペレーティングシステムのインストール、ネットワークアドレスの割り当て、ストレージデバイスの設定などについての知識を含む、ソリューションをインストールするオペレーティングシステムプラットフォームについての十分な知識。
- Java ES インストーラについての一般的知識。詳細については、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「Java ES インストーラのしくみ」を参照してください。
- LDAP ディレクトリについての一般的知識。
- ソリューションに必要なディスク領域を予測するための、ハードウェアに関する十分な知識。

インストール計画の作成は、複数の人物が分担して行わなければならない場合があります。たとえば、計画の主要責任者は、ソリューションのインストールおよび設定のために必要な情報を作成するために、LDAP に詳しい人物に問い合わせる必要が生じる場合があります。

内容の紹介

第 1 章では、インストール計画プロセスの概要を示します。

第 2 章では、配備アーキテクチャーに含まれていないが、Java ES ソリューションをインストールするために必要な追加の情報を作成する方法について説明します。

第 3 章では、インストール計画全般について説明したあと、特定の Java ES ソリューション用のインストール計画を作成する方法について説明します。

Java ES マニュアルセット

Java ES ドキュメントセットには、配備計画およびシステムインストールに関する情報が記載されています。システムマニュアルの URL は `http://docs.sun.com/coll/1286.2` です。Java ES の概要を理解するため、次の表に紹介されているマニュアルを、記載されている順番に参照してください。

表 P–1　Java Enterprise System のマニュアル

マニュアルタイトル	内容
『Sun Java Enterprise System 5 リリースノート (UNIX 版)』	既知の問題など、Java ES に関する最新の情報が記載されています。これに加えて、各コンポーネントには、個別のリリースノートもあります。

表 P–1　Java Enterprise System のマニュアル　*(続き)*

マニュアルタイトル	内容
『Sun Java Enterprise System 5 技術の概要』	Java ES の技術的および概念的な基礎について説明します。コンポーネント、アーキテクチャー、プロセス、および機能について説明しています。
『Sun Java Enterprise System Deployment Planning Guide 』	Java ES に基づく企業配備ソリューションの計画および設計に関する情報を提供します。配備の計画および設計に関する基本的概念と原則を示し、ソリューションのライフサイクルについて説明し、Java ES に基づくソリューションを計画する際に使用する高度な例と戦略を提供します。
『Sun Java Enterprise System 5 インストール計画ガイド』	Java ES の配備に関し、ハードウェア、オペレーティングシステム、およびネットワーク面の実装仕様の開発に役立つ情報を提供します。インストールおよび設定計画を遂行する上で注意すべきコンポーネントの依存関係などの問題について説明します。
『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』	Solaris オペレーティングシステムまたは Linux オペレーティングシステムに対する Java ES のインストールシーケンスについて説明します。また、インストール後にコンポーネントを設定する方法、および設定したコンポーネントが正常に機能するかどうかを確認する方法についても説明します。
『Sun Java Enterprise System 5 インストールリファレンス (UNIX 版)』	設定パラメータに関する追加情報や、設定計画で使用するワークシート、デフォルトディレクトリやポート番号などの参考情報を提供します。
『Sun Java Enterprise System 5 アップグレードガイド (UNIX 版)』	Solaris オペレーティングシステムまたは Linux オペレーティング環境で Java ES をアップグレードする方法および手順について説明します。
『Sun Java Enterprise System 5 監視ガイド (UNIX 版)』	各製品での Monitoring Framework の設定方法と、Monitoring Console を使用してリアルタイムデータを参照し、しきい値のアラームを設定する方法を示します。
『Sun Java Enterprise System 用語集』	Java ES ドキュメントで使用される用語について説明します。

関連マニュアル

インストール計画の作成に役立つ可能性の高いマニュアルを次に示します。

- 『Sun Java Enterprise System 5 技術の概要』では、Java ES のコンポーネントと、それらのコンポーネントが提供するサービスについて説明します。
- 『Sun Java Enterprise System Deployment Planning Guide』では、ビジネスのニーズを分析して配備アーキテクチャーを作成する方法について説明します。
- 『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』では、Java ES インストーラの操作方法について説明します。

- 『Sun Java Enterprise System 5 インストールリファレンス (UNIX 版)』では、Java ES インストーラの入力値の完全な一覧を示します。
- 『Sun Java Communications Suite 5 配備計画ガイド』など、個別コンポーネントの配備計画ガイドでは、コンポーネントの設定についての詳細な情報を示しています。
- このマニュアルセットで使用されている用語の一覧については、『Sun Java Enterprise System 用語集』を参照してください。

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P–2 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 machine_name% you have mail.
AaBbCc123	ユーザーが入力する文字を、画面上のコンピュータ出力と区別して示します。	machine_name% **su** Password:
aabbcc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *filename* と入力します。
『』	参照する書名を示します。	『コードマネージャー・ユーザーズガイド』を参照してください。
「」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

コード例は次のように表示されます。

- C シェル

```
machine_name% command y|n [filename]
```

- C シェルのスーパーユーザー

```
machine_name# command y|n [filename]
```

- Bourne シェルおよび Korn シェル

```
$ command y|n [filename]
```

- Bourne シェルおよび Korn シェルのスーパーユーザー

```
# command y|n [filename]
```

[ ] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

コマンド例のシェルプロンプト

次の表は、デフォルトのシステムプロンプトとスーパーユーザープロンプトを示しています。

表 P–3 シェルプロンプト

シェル	プロンプト
UNIX および Linux システムの C シェル	machine_name%
UNIX および Linux システムの C シェルのスーパーユーザー	machine_name#
UNIX および Linux システムの Bourne シェルおよび Korn シェル	$
UNIX および Linux システムの Bourne シェルおよび Korn シェルのスーパーユーザー	#
Microsoft Windows のコマンド行	C:\

記号の規則

次の表は、この用語集で使用される記号の一覧です。

表 P–4 記号の規則

記号	説明	例	意味
[ ]	省略可能な引数やコマンドオプションが含まれます。	ls [-l]	-l オプションは必須ではありません。
{ | }	必須のコマンドオプションの選択肢を囲みます。	-d {y|n}	-d オプションには y 引数か n 引数のいずれかを使用する必要があります。
${ }	変数参照を示します。	${com.sun.javaRoot}	com.sun.javaRoot 変数の値を参照します。
-	同時に押すキーを示します。	Control-A	Control キーを押しながら A キーを押します。
+	順番に押すキーを示します。	Ctrl+A+N	Control キーを押してから放し、それに続くキーを押します。
→	グラフィカルユーザーインタフェースでのメニュー項目の選択順序を示します。	「ファイル」→「新規」→「テンプレート」	「ファイル」メニューから「新規」を選択します。「新規」サブメニューから「テンプレート」を選択します。

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	内容
マニュアル	http://jp.sun.com/documentation/	PDF 文書および HTML 文書をダウンロードできます。
サポートおよびトレーニング	http://jp.sun.com/supportraining/	技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。

◆ ◆ ◆ 第 1 章

1 インストール計画の概要

『Sun Java Enterprise System 5 インストール計画ガイド』では、Sun Java™ Enterprise System (Java ES) ソフトウェアのインストールを準備する方法について説明します。この章では、この『インストール計画ガイド』で扱う範囲と内容を定義します。この章の内容は次のとおりです。

- 11 ページの「このリリースで使用される Java ES コンポーネント」
- 12 ページの「Java ES インストールの定義」
- 13 ページの「インストール計画のタスク」

このリリースで使用される Java ES コンポーネント

Java ES 5 リリースには、次のコンポーネントが含まれています。これらのコンポーネントは、インストール時に個別に選択が可能です。

このマニュアルで使用されている代替名または省略名は、コンポーネント名とバージョンのあとに括弧で囲まれています。

- Access Manager 7.1
- Application Server 8.2 Enterprise Edition およびパッチ (Application Server)
- Directory Preparation Tool 6.4
- Directory Proxy Server 6.0
- Directory Server Enterprise Edition 6.0 (Directory Server)
- High Availability Session Store 4.4 (HADB)
- Java DB 10.1
- Message Queue 3.7 UR1
- Monitoring Console 1.0
- Portal Server 7.1
- Portal Server Secure Remote Access 7.1
- Service Registry 3.1
- Sun Cluster 3.1 8/05 (Sun Cluster ソフトウェア)
- Sun Cluster Agents 3.1

- Sun Cluster Geographic Edition 3.1 2006Q4 (Sun Cluster Geographic)
- Web Proxy Server 4.0.4
- Web Server 7.0

注 – HP-UX では、Sun Cluster コンポーネント、Directory Preparation Tool、HADB、またはサードパーティーの Web コンテナはサポートされません。Linux では、Sun Cluster コンポーネントはサポートされず、IBM WebSphere のサードパーティーのコンテナのみサポートされます。

注 – このマニュアルでは、Sun Java System Communications Suite のコンポーネントも参照します。これらのコンポーネントは、たいていは Java ES コンポーネントとともに配備されます。

注 – Directory Preparation Tool は Communications 製品だけで使用され、便利なツールとして Java ES リリースの Directory Server に同梱されています。Directory Preparation Tool の詳細については、『Sun Java Communications Suite 5 インストールガイド』の第 8 章「Directory Preparation Tool (comm_dssetup.pl) 」を参照してください。

Java ES インストールの定義

このマニュアルで説明するインストールプロセスには、Java ES ソリューションのインストールと基本設定が含まれます。このマニュアルで使用する「インストール」という用語は、Java ES インストーラを使用して、Java ES のコンポーネント群をコンピュータシステムにコピーすることを意味します。このマニュアルで使用する「設定」という用語は、インストーラまたは設定ウィザードを使用して、Java ES コンポーネントのインスタンスを設定することを意味します。この基本設定を実行したあと、インスタンスを起動し、その基本動作を検証し、インスタンスがソリューション内のほかのコンポーネント群と正しく相互動作するかどうかを確認することができます。

このマニュアルで説明する基本設定は、コンポーネントの機能の一部しか網羅していません。たとえば、Portal Server の基本インストールおよび設定では、Access Manager や Directory Server などのほかのコンポーネントと相互動作するインスタンスを作成します。基本設定では、基本ポータルデスクトップへのコンテンツの追加など、Portal Server の機能のほかの側面については扱いません。基本設定の範囲を超えるコンポーネントの機能を設定するには、『Sun Java System Portal Server 7.1 Configuration Guide 』などの、個別のコンポーネントのマニュアルを参照する必要があります。

インストール計画のタスク

次の表は、インストール計画のタスクのうち、すべての Java ES ソリューションに共通するものの一覧です。左の列には、上位レベルのタスクと、各タスクのサブタスクの一覧を示しています。右の列には各タスクを実行する手順の参照先を示しています。

表 1–1　インストール計画のタスク

作業	情報の参照先
1. 実装の仕様を作成する	第 2 章
配備アーキテクチャーを分析する	15 ページの「配備アーキテクチャーの分析」
ネットワーク接続仕様を作成する	18 ページの「ネットワーク接続仕様の作成」
コンピュータハードウェア仕様およびオペレーティングシステム仕様を作成する	17 ページの「コンピュータハードウェアおよびオペレーティングシステム仕様の作成」
ユーザー管理仕様を作成する	20 ページの「ユーザー管理仕様の作成」
2. インストールと設定の問題について理解する	25 ページの「インストール計画の問題」
分散インストールがインストール計画に及ぼす影響を理解する	26 ページの「分散インストール」
コンポーネントの相互動作の設定がインストール計画の中で占める役割について理解する	32 ページの「相互動作のための設定」
コンポーネントの依存性がインストール計画に及ぼす影響を理解する	26 ページの「コンポーネントの依存関係」
ソリューションで使用する冗長性戦略がインストール計画に及ぼす影響を理解する	33 ページの「冗長性戦略」
LDAP ディレクトリの問題がインストール計画に及ぼす影響を理解する	34 ページの「LDAP スキーマと LDAP ディレクトリツリーの構造」
インストーラの動作モードがインストール計画に及ぼす影響を理解する	35 ページの「Java ES インストーラの動作」
3. インストール計画を作成する	40 ページの「インストール計画の作成」
コンポーネントインスタンスのインスタンスと設定の適切な順序を決定する	26 ページの「コンポーネントの依存関係」
各コンポーネントインスタンスに固有の入力値を決定する	32 ページの「相互動作のための設定」

インストール計画のタスクは、このマニュアルで説明する方法に従って正しい順序で実行することが重要です。

◆ ◆ ◆ 第 2 章

2 実装の仕様の作成

配備アーキテクチャーは、Java ES ソリューションの上位レベルの技術的記述であり、ソリューションをインストールおよび設定するために必要なすべての情報が含まれているわけではありません。この章では、配備アーキテクチャーを分析し、一連の実装仕様を作成する手順について説明します。実装仕様を作成するのは、ソリューションをインストールおよび設定するために必要な補足情報の作成作業を準備するためです。

この章では、実装仕様に関する次の内容を説明します。

- 15 ページの「配備アーキテクチャーの分析」
- 17 ページの「コンピュータハードウェアおよびオペレーティングシステム仕様の作成」
- 18 ページの「ネットワーク接続仕様の作成」
- 20 ページの「ユーザー管理仕様の作成」

配備アーキテクチャーの分析

図 2–1 は、一般的な配備アーキテクチャーの例です。この配備アーキテクチャーは、ポータルと通信サービスを提供する Java ES ソリューションを定義します。このアーキテクチャーでは、Access Manager を使用して、通信サービスへのシングルサインオンを実現しています。また、Portal Server と Communications Express の両方を使用して、メッセージングサービスとカレンダーサービスをエンドユーザーに提供しています。このアーキテクチャーには、Communications Suite のコンポーネントが含まれています。

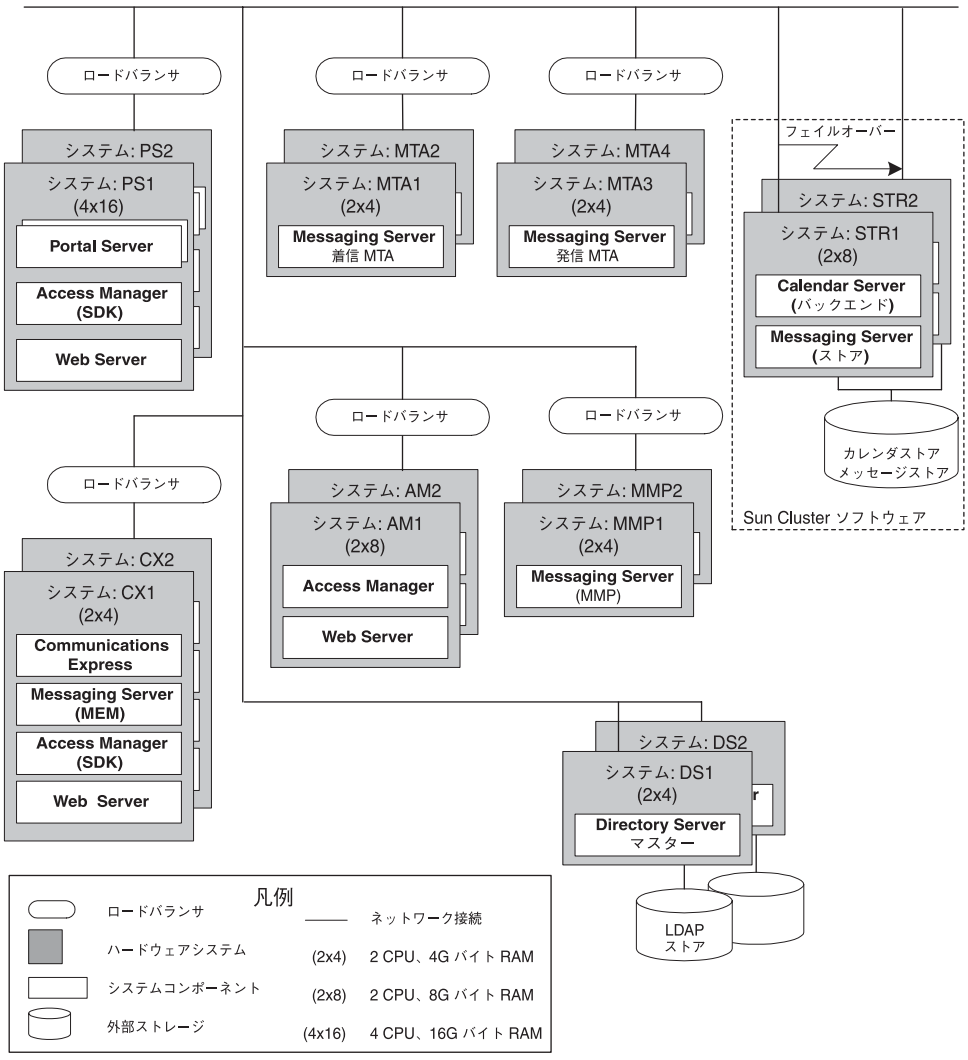

図 2–1 配備アーキテクチャーの例

図 2–1 には、次のものを含めて、ソリューションについての多くの情報が含まれます。

- ソリューションで使用されるコンピュータの数
- 各コンピュータに必要な CPU 数と RAM 容量
- 各コンピュータにインストールされるコンポーネントインスタンス
- ソリューションで使用される各コンポーネントのインスタンスの数

- サービス品質要件を満たすためにソリューションで使用される冗長性戦略 (負荷分散、Directory Server マルチマスターレプリケーション、および Sun Cluster 技術)
- サービス品質要件を満たすためのもう 1 つの手法である、Messaging Server サブコンポーネントの分散インストール

配備アーキテクチャー例のこれらの特性は、ソリューションがどのようにインストールおよび設定されるかに影響します。インストールの計画を始めるにあたり、まず同じようにして配備アーキテクチャーを分析し、何台のコンピュータシステムを監視するか、各コンピュータシステムに何個のコンポーネントインスタンスをインストールするか、どの冗長性戦略を利用するか、などを決定します。

コンピュータハードウェアおよびオペレーティングシステム仕様の作成

配備アーキテクチャーで定義される情報に加えて、ソリューションで使用する各コンピュータのオペレーティングシステムを指定する必要があります。また、インストール先のハードウェアに関して追加の情報を作成する必要もあります。そのための決定はサービス品質要件に基づいて行われ、サービス品質要件を満たすために必要なハードウェアおよびオペレーティングシステムに関する最善の推測に相当します。

図 2–1 で示した配備アーキテクチャーのサービス品質要件を満たすために、表 2–1 のオペレーティングシステム仕様およびコンピュータハードウェア仕様が作成されました。

表 2–1 サンプル配備アーキテクチャーのコンピュータハードウェア/OS 仕様

コンピュータシステム	ハードウェアモデル	CPU の数	RAM (G バイト)	ディスク数	オペレーティングシステム
mscs01	Sun Fire V440 Server	4	16	4	Solaris 9
mscs02					
commx01	Sun Fire V240 Server	2	4	2	Solaris 10
commx02				4	
ds01	Sun Fire V240 Server	2	8	4	Solaris 10
ds02					
am01	Sun Fire V240 Server	2	8	4	Solaris 10
am02					

表 2–1 サンプル配備アーキテクチャーのコンピュータハードウェア/OS 仕様 (続き)

コンピュータシステム	ハードウェアモデル	CPU の数	RAM (G バイト)	ディスク数	オペレーティングシステム
ms-mmp01 ms-mmp02	Sun Fire V240 Server	2	4	2	Solaris 10
ms-mtai01 ms-mtai02	Sun Fire V240 Server	2	4	2	Solaris 10
ms-mtao01 ms-mtao02	Sun Fire V240 Server	2	4	2	Solaris 10
ps01 ps02	Sun Fire V440 Server	4	16	4	Solaris 10
protect	Sun Fire V240	2	4	2	Solaris 10

ソリューションで使用するコンピュータシステムについても、同様の情報を作成する必要があります。

ヒント – コンピュータハードウェア/OS 仕様が完成すれば、コンピュータシステムを設定できるようになります。メモリーやディスクドライブをインストールし、オペレーティングシステムをインストールすることで、Java ES コンポーネントをシステムにインストールできるようになります。

ネットワーク接続仕様の作成

配備アーキテクチャーには、ソリューションで使用するすべてのハードウェアを相互接続するために必要な情報の多くが含まれます。ネットワークを接続するために必要な追加情報の作成を支援するために、図 2–2 の例のようなネットワーク接続仕様を準備する必要があります。

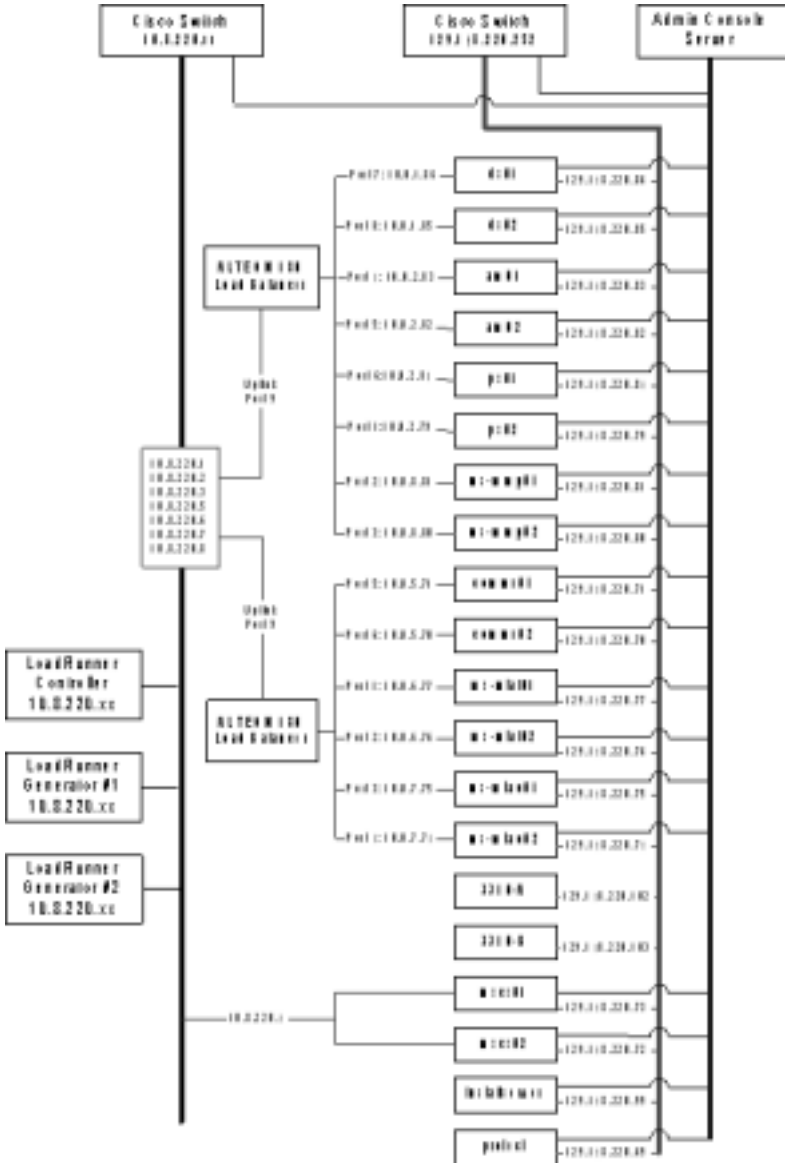

図2–2　ネットワーク接続仕様の例

配備アーキテクチャー例のネットワーク接続仕様では、配備アーキテクチャー図にはない次の情報を追加しています。

- ソリューションで使用するすべてのコンピュータおよびハードウェアロードバランサの IP アドレス

- コンピュータをロードバランサに接続するために使われるロードバランサのポート番号
- ロードバランサの IP アドレスは、負荷分散対象のコンピュータによって提供されるサービスにアクセスするために使われる論理アドレスを示す

ソリューションに必要な接続についても、同様の情報を作成する必要があります。

ヒント– ネットワーク接続仕様が完成したら、ネットワークが接続可能になり、Java ES コンポーネントのインストールと設定の準備ができたことになります。

ユーザー管理仕様の作成

Java ES ソリューションのインストールおよび設定のプロセスによって、LDAP ディレクトリが設定されます。Java ES をインストールおよび設定すると LDAP スキーマと LDAP ディレクトリツリーの両方が作成されます。スキーマとディレクトリツリーの詳細は、インストールと設定のプロセスで入力する値によって決定されます。このため、インストール計画には、Java ES ソリューションをサポートするスキーマおよびディレクトリ構造の仕様を作成することが含まれます。

ディレクトリツリー構造とスキーマは、ソリューションが提供するサービスに対応している必要があります。ここでは、利用可能なオプションの基本的な説明と、各オプションがサポートするサービスを示します。ただし、この節の主な目的は、Java ES ソリューションをサポートするスキーマおよびディレクトリツリー構造を作成するために、Java ES インストーラおよび Java ES 設定ツールのための入力値を選択する方法について説明することです。

スキーマの選択およびディレクトリツリーの設計については、『Sun Java System Directory Server Enterprise Edition 6.0 Deployment Planning Guide』などの追加資料を参照してください。

ソリューションの **LDAP** スキーマの指定

Directory Server を使用する Java ES ソリューションは、スキーマ 1 およびスキーマ 2 として知られる、標準 LDAP スキーマの 2 つのバージョンのいずれかを使用できます。ユーザー管理仕様では、ソリューションでスキーマ 1 とスキーマ 2 のどちらを使用するかを指定します。

スキーマ 2 は、Access Manager の使用と、Access Manager のシングルサインオン機能をサポートします。Access Manager を使用するソリューションでは、スキーマ 2 を使用する必要があります。

インストールプロセスでは、指定されたスキーマのディレクトリを次のように設定します。

- スキーマ 1 ディレクトリを確立するには、単に Directory Server をインストールします。スキーマ 1 はデフォルトのスキーマバージョンです。
- スキーマ 2 ディレクトリを確立するには、Directory Server および Access Manager をインストールします。Access Manager をインストールすると、ディレクトリが変更され、そのディレクトリがスキーマ 2 ディレクトリに変換されます。

ヒント – 1 回のインストーラセッションで Directory Server と Access Manager を 1 台のコンピュータにインストールすると、ディレクトリはスキーマ 2 用に設定されます。

分散構成のソリューションでは、まず 1 台のコンピュータに Directory Server をインストールします。次に、2 台目のコンピュータに Access Manager をインストールします。Access Manager をインストールするときは、リモートコンピュータ上の既存のディレクトリを指定するため、ディレクトリのスキーマはスキーマ 2 に設定されます。

ソリューションによっては、スキーマを拡張するための次の手順が必要な場合があります。

- ソリューションで Communications Suite (Messaging Server または Calendar Server、あるいはその両方) を使用する場合、インストールプロセスで Directory Preparation Tool を使用して、いくつかの追加スキーマ拡張を適用する必要があります。これらの拡張は、Messaging Server または Calendar Server がインストールされる前に適用されます。それらの拡張はスキーマ 1 ディレクトリまたはスキーマ 2 ディレクトリのどちらかに適用できます。Directory Preparation Tool のための指示が含まれるインストール計画の例については、『Sun Java Enterprise System 2005Q4 Deployment Example: Telecommunications Provider Scenario 』を参照してください。
- ソリューションでスキーマ 2 を使用する場合、インストールプロセスでは、メッセージングサービスおよびカレンダサービスに必要な Access Manager による認証と承認をサポートするために、Delegated Administrator によるいくつかの追加スキーマ拡張を適用する必要があります。これらのスキーマ拡張が適用されるインストール計画の例については、『Sun Java Enterprise System 2005Q4 Deployment Example: Telecommunications Provider Scenario 』を参照してください。

LDAP スキーマ仕様は、ソリューションで使用されるスキーマと、ソリューションによって必要とされるすべてのスキーマ拡張を識別します。

ソリューションのディレクトリツリー構造の指定

Java ES ソリューションの LDAP ディレクトリは、ソリューションでユーザーデータを組織化する必要性に応じて、簡単な場合もあれば複雑な場合もあります。LDAP ディレクトリはその性質上、構造的には柔軟です。Java ES ではディレクトリの構造

は特に定められていませんが、ディレクトリツリー構造がインストールと設定のプロセスを通じて実装されます。ディレクトリツリーは、インストールと設定のプロセスの開始前に設計する必要があります。

インストールと設定のプロセスでは、次のようにしてディレクトリツリー構造が確立されます。

1. インストーラを実行して Directory Server をインストールするためには、ディレクトリのベースサフィックス (ルートサフィックスまたはルート DN とも呼ばれる) の入力値が必要です。Java ES インストーラは、入力値を使用してディレクトリのベースサフィックスを確立します。ディレクトリツリーにベースサフィックス名を指定する必要があります。

   ヒント – Messaging Server または Calendar Server を使用しない、ディレクトリツリーが単純なソリューションでは、ユーザーおよびグループのデータをベースサフィックスの直下に保存できます。

   ---

2. Messaging Server (Communications Suite コンポーネント) の設定ウィザードを実行して Messaging Server インスタンスを作成するためには、LDAP 組織 DN の入力値が必要です。設定ウィザードによってディレクトリツリーが分岐され、ウィザードでの DN 入力を使用して LDAP 組織が作成されます。この組織は、Messaging Server インスタンスによって管理される電子メールドメインを表します。ウィザードでは、ユーザーおよびグループデータ用に Messaging Server インスタンスが使用する電子メールドメイン組織も設定します。インストール計画には、電子メールドメイン組織の DN も含まれます。このプロセスによって作成されるディレクトリツリー構造の例については、図 2–3 を参照してください。この例では、インストーラによって作成されるベースサフィックスは `o=examplecorp` です。Messaging Server の設定ウィザードによって作成される電子メールドメイン組織は、`o=examplecorp.com,o=examplecorp` です。
3. Calendar Server、Communications Express、Instant Messaging、および Delegated Administrator (Communicaitons Suite コンポーネント) の設定ウィザードでは、LDAP DN の入力値が必要となります (ウィザードでの表示名は異なる場合がある)。ソリューションでシングルサインオンを使用する場合、すべての設定ウィザードで同じ値が入力されます。入力値は、Messaging Server のウィザードによって作成される電子メールドメイン組織です。この設定では、すべてのコンポーネントがユーザーデータの保存および検索場所として同じ LDAP 組織を使用する、という結果になります。ユーザーに関するすべての情報を単一のディレクトリエントリに保存でき、Access Manager のシングルサインオン機能を使用できます。

図 2–3 は、このプロセスによって作成されるディレクトリツリー構造の例を示したものです。この例では、Java ES インストーラによってベースサフィックス `o=examplecorp` が確立され、Messaging Server の設定ウィザードによって組織 `o=examplecorp.com,o=examplecorp` が追加されました。この組織は、`examplecorp.com`

という名前の電子メールドメインを表します。電子メールドメインのユーザーデータは `ou=people,o=examplecorp.com,o=examplecorp` に保存されます。ソリューション内のほかの Java ES コンポーネントも、ユーザーデータを ou=people,o=examplecorp.com,o=examplecorp から検索するように設定されます。

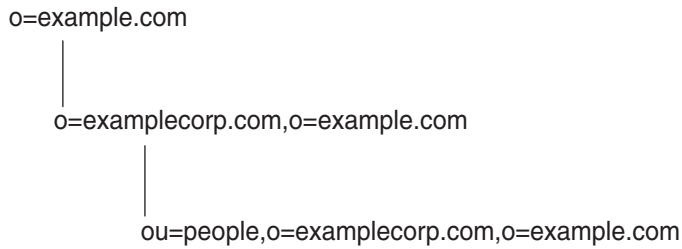

図 2–3　LDAP ディレクトリツリーの例

ソリューションで図 2–3 に示したディレクトリツリーが必要な場合は、ベースサフィックスの名前と、電子メールドメインを表す組織がユーザー管理仕様に追加されます。

ディレクトリツリーの例には 1 つのメールドメインのみが含まれます。多くのソリューションでは、ユーザーデータを組織化するために、これよりも複雑なツリーが必要です。同じインストールと設定のための基本手順で、より複雑なディレクトリ構造を確立できます。たとえば、ソリューションで必要な場合には、複数の電子メールドメインをサポートするようにディレクトリを設定できます。

複数の電子メールドメインを確立するには、Messaging Server の複数のインスタンスを設定します。各インスタンスは 1 つの電子メールドメインを管理します。例については、『Sun Java Enterprise System 2005Q4 Deployment Example: Telecommunications Provider Scenario 』を参照してください。

ソリューションがディレクトリとの対話に Access Manager を使用する場合、Java ES ソリューションでほかの LDAP ディレクトリを使用することが可能です。使用するディレクトリサーバーは、LDAP バージョン 3 (LDAP v3) 準拠のディレクトリサーバーである必要があります。

◆ ◆ ◆ 第 3 章

3 インストール計画の準備

第 2 章の説明に従って実装仕様を作成すると、インストール計画を準備するために必要な情報が用意されます。インストール計画では、Java ES ソリューションのインストールと設定に必要なすべての手順を記述します。インストール計画では、特定の Java ES ソリューションの実装に必要なすべての手順を記述します。

この章では、インストール計画を準備する方法について説明します。最初に、配備アーキテクチャーおよび実装仕様内の情報の定義を行います。ここには、Java ES ソリューションの配備状態を記述します。これらのドキュメントに記載されている情報を分析し、Java ES インストーラと設定ウィザードを使用して仕様書に記述したソリューションを実装する方法を決定します。

この章では、インストール計画の作成方法について説明します。この章で扱う内容は次のとおりです。

- 25 ページの「インストール計画の問題」
- 40 ページの「インストール計画の作成」

インストール計画の問題

インストールと設定のプロセスの目標は、配備アーキテクチャーに記述された分散システムを実現することです。分散システムは、複数のコンピュータ上で実行され、互いに連携して動作するコンポーネントインスタンスで構成されます。正しく機能する分散システムを構築するには、複数のコンピュータにコンポーネントインスタンスをインストールし、コンポーネントインスタンス間の相互動作を確立する基本設定を実行する必要があります。

インストールと設定の手順は、Java ES インストーラの動作と、個別のコンポーネントの要件によって決定されます。正しく機能する分散システムを確実に実現するためには、インストーラを適切に使用する、また、ソリューションで使用されるコンポーネントの要件を考慮したインストール計画を作成する必要があります。計画では、それぞれのコンポーネントインスタンスのインストールと基本設定の実行の正

しい順序を記述する必要があります。計画ではさらに、コンポーネントインスタンスを相互動作させるための設定値を指定する必要もあります。

この節では、インストール計画の作成時に考慮する必要のある主な問題について説明します。

分散インストール

本稼働 Java ES ソリューションに求められるサービス品質要件は、結果として複数のコンピュータにコンポーネントインスタンスを配置するアーキテクチャーになります。たとえば、信頼性のあるポータルサービスを実現するために、2 台の異なるコンピュータに Portal Server の 2 つのインスタンスを配置し、負荷分散を使用して 2 つのインスタンス間にフェイルオーバー関係を確立するアーキテクチャーが必要になる場合があります。

ただし、Java ES インストーラは一度に 1 台のコンピュータ上でしか動作しません。したがって、分散ソリューションをインストールするときは、ソリューションで使用されるすべてのコンピュータ上で個別にインストーラを実行する必要があります。

多くの場合、1 台のコンピュータに 1 つまたは複数のコンポーネントをインストールしたあと、設定ウィザードを実行して基本設定を実行します。通常は、1 台のコンピュータ上でインストールと設定を完了してから、別のコンピュータにコンポーネントの別のセットをインストールして設定する作業に進みます。分散コンポーネントインスタンスをインストールして設定するために、図 3–1 に示すような一連のタスクを実行する場合があります。

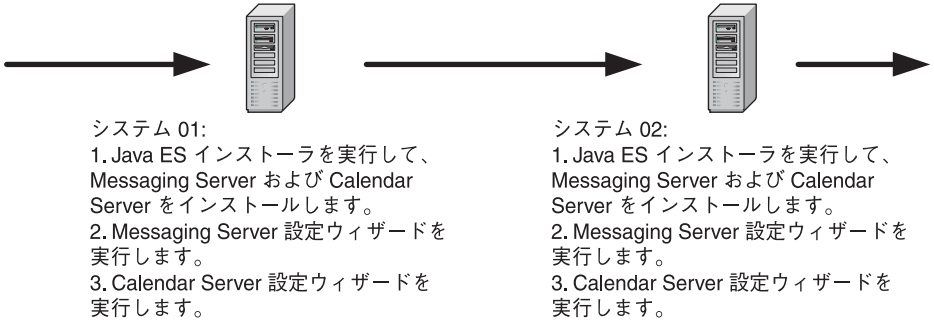

図 3–1 分散インストール手順の例

コンポーネントの依存関係

Java ES コンポーネントの中には、ほかのコンポーネントをまずインストールして設定してからでないと、インストールと設定ができないものがあります。依存性が発生する理由はいくつかあります。

- 一部のコンポーネントは、ほかの特定のコンポーネントがインストールおよび設定されていないと正しく機能しません。たとえば、Access Manager を適切に動作させるためには、LDAP ディレクトリによって提供される、ユーザーとサービスについての情報にアクセス可能である必要があります。Access Manager のインストールと設定の手順では、Access Manager が、すでに動作しているディレクトリサービスと相互動作するために URL の入力が必要です。この依存性のため、Directory Server をインストールおよび設定してから、Access Manager をインストールおよび設定する必要があります。
- 一部のコンポーネントは、既存のコンポーネントの設定を変更します。たとえば、Access Manager をインストールして設定すると、LDAP ディレクトリスキーマが変更されます。ソリューションで Access Manager を使用する場合、Access Manager のインストールよりも先に LDAP ディレクトリをインストールして設定することを指定する必要があります。
- 多くの Java ES コンポーネントは Web アプリケーションです。これらのコンポーネントは、正しく機能させるためには Web コンテナに配備する必要があります。Web アプリケーションコンポーネントをインストールして設定する前に、Web コンテナのインストールと起動を計画する必要があります。Web Server、Application Server、または一部のサードパーティー製 Web コンテナを使用できますが、Web アプリケーションコンポーネントをインストールする時点で Web コンテナがそのコンピュータ上に存在するように計画する必要があります。

  ヒント – ソリューションで Web Server または Application Server を使用する場合、Java ES インストーラは Web コンテナと Web アプリケーションコンポーネントを同時にインストールでき、Web アプリケーションコンポーネントを Web コンテナに自動的に配備できます。

- アーキテクチャーによっては、Sun Cluster ソフトウェアによって提供される高可用性クラスタにコンポーネントをインストールすることが要求されることがあります。そのようなコンポーネントをインストールして設定する前に、Sun Cluster ソフトウェアをインストールして実行させておく必要があります。また、該当するコンポーネント用の Sun Cluster エージェントをインストールして設定しておく必要もあります。

これらの依存性には、ソリューションレベルのものとローカルのものがあることに注意してください。インストール計画を作成するときは、ソリューションレベルの依存性とローカルの依存性を分けて考えます。次の例で違いを説明します。

Directory Server に対する Access Manager の依存性はソリューションレベルの依存性です。Access Manager をインストールするとき、Directory Server の 1 つまたは複数のインスタンスによって提供されるディレクトリサービスの URL を指定します。Directory Server のインストールと設定が完了すると、ソリューション内のすべてのコンポーネントで利用できるディレクトリサービスが提供されます。このタイプの依存性は、コンポーネントインスタンスのインストールと設定に関するソリュー

ションレベルのシーケンスを決定します。Directory Server は、Access Manager よりも先にインストールし設定する必要があります。インストール計画において、ソリューションレベルの依存性は、インストールと設定のステップの全体的なシーケンスを決定します。まず Directory Server をインストールし、そのあと、Access Manager などの、ディレクトリサービスに依存するコンポーネントを追加するよう計画できます。

Web コンテナに対する Access Manager の依存性はローカル依存性です。この依存性を満たすには、Access Manager を実行するコンピュータに Web コンテナをインストールする必要があります。ただしこの Web コンテナは、ソリューション全体に対して Web コンテナを提供するものではありません。分散アーキテクチャーで Access Manager とは異なるコンピュータ上に Portal Server をインストールすると指定されている場合は、両方のコンピュータに Web コンテナをインストールするよう計画する必要があります。各 Web コンテナは、それぞれ異なるコンポーネントをローカルでサポートします。したがって分散ソリューションでは、ソリューション全体にサービスを提供する Web コンテナの 1 つの決まった場所は存在しないため、インストールシーケンス全体を通して何回か Web コンテナをインストールするよう計画する必要があります。

ソリューションのインストール計画を作成するために、ソリューションを記述してコンポーネント間の依存性を識別する配備アーキテクチャーを分析します。計画では、すべての依存性を満たすシーケンスでコンポーネントをインストールおよび設定する必要があります。一般に、全体的なインストールシーケンスはソリューションレベルの依存性を基にして作成します。その後、それぞれのコンピュータ上に存在する可能性があるローカルの依存性を考慮します。

コンポーネントの依存性のリストを表 3–1 に示します。これらの依存性への対処方法については、40 ページの「インストール計画の作成」の個別コンポーネントの説明を参照してください。

表 3–1　Java ES コンポーネントの依存性

製品コンポーネント	依存性	依存性の性質	ローカルである必要性
Access Manager	Directory Server	設定データを格納するため。ユーザーデータを格納し、検索を可能にするため	不可

表 3–1 Java ES コンポーネントの依存性 (続き)

製品コンポーネント	依存性	依存性の性質	ローカルである必要性
	J2EE Web コンテナ、次のいずれか -Application Server -Web Server -BEA WebLogic Server -IBM WebSphere Application Server	Access Manager はこれらの Web コンテナのいずれかに配備される必要がある	可能
Access Manager SDK	Access Manager	基本的な Access Manager サービスを提供するため	不可
	J2EE Web コンテナ、次のいずれか -Application Server -Web Server -BEA WebLogic Server -IBM WebSphere Application Server	Access Manager SDK はこれらの Web コンテナのいずれかに配備される必要がある	可能
Access Manager Distributed Authentication	Access Manager	基本的な Access Manager サービスを提供するため	不可
	J2EE Web コンテナ、次のいずれか -Application Server -Web Server -BEA WebLogic Server -IBM WebSphere Application Server	Access Manager SDK はこれらの Web コンテナのいずれかに配備される必要がある	可能
Access Manager セッションフェイルオーバー	Access Manager	基本的な Access Manager サービスを提供するため	不可
	Message Queue	信頼性のある非同期メッセージングを提供するため	不可
Application Server	Message Queue	信頼性のある非同期メッセージングを提供するため	可能
	Web Server (オプション)	Application Server インスタンス間の負荷分散を提供するため	可能

表 3–1 Java ES コンポーネントの依存性 (続き)

製品コンポーネント	依存性	依存性の性質	ローカルである必要性
	High Availability Session Store (オプション)	Application Server 間のフェイルオーバーをサポートするセッション状態を格納するため	可能
Directory Proxy Server	Directory Server	基盤の LDAP ディレクトリサービスを提供するため	不可
Directory Server	なし		
High Availability Session Store	なし		
Java DB	なし		
Message Queue	Directory Server (省略可能)	管理されたオブジェクトと持続メッセージを格納するため	不可
	J2EE Web コンテナ (次のいずれか、省略可能): -Application Server -Web Server	クライアントとメッセージブローカ間の HTTP トランスポートをサポートするため	不可
	Sun Cluster (省略可能)	高可用性ソリューションでの Message Queue の使用をサポートするため	不可
Portal Server	J2EE Web コンテナ (次のいずれか): -Application Server -Web Server -BEA WebLogic Server -IBM WebSphere Application Server	Portal Server はこれらの Web コンテナのいずれかに配備される必要がある	可能
	Directory Server	認証と承認に使われるユーザーデータを格納するため	不可
	Access Manager または Access Manager SDK	Access Manager サービスを提供するため。ローカルの Access Manager SDK はリモートの Access Manager へのアクセスを提供する	可能

表 3–1 Java ES コンポーネントの依存性 (続き)

製品コンポーネント	依存性	依存性の性質	ローカルである必要性
	Service Registry Client	コンパイルのために必要なライブラリを提供するため	不可
Portal Server Secure Remote Access	Portal Server	基盤のポータルサービスを提供するため	不可
	Access Manager または Access Manager SDK のどちらか	Access Manager サービスを提供するため。ローカルの Access Manager SDK はリモートの Access Manager へのアクセスを提供する	可能
Rewriter プロキシ	Portal Server	基盤のポータルサービスを提供するため	不可
Netlet プロキシ	Portal Server	基盤のポータルサービスを提供するため	不可
Service Registry	Application Server	必要なコンテナサービスを提供するため	可能
	Service Registry クライアント	必要なクライアントインタフェースを提供するため	可能
Service Registry クライアント	なし		
Sun Cluster ソフトウェア	なし		
Sun Cluster Agents	Sun Cluster	基本的なクラスタサービスを提供するため	可能
Sun Cluster Geographic Edition	Sun Cluster	基本的なクラスタサービスを提供するため	可能
Web Proxy Server	Web Server	Web Server で実行する Web アプリケーションへのリモートアクセスを提供するため	可能
	Directory Server (省略可能)	認証と承認に使われるユーザーデータを格納するため	不可

表 3–1 Java ES コンポーネントの依存性 (続き)

製品コンポーネント	依存性	依存性の性質	ローカルである必要性
Web Server	Directory Server (省略可能)	認証と承認に使われるユーザーデータを格納するため	不可

相互動作のための設定

インストールと設定のプロセスの目標は、複数のコンポーネントインスタンスが相互に連携して動作するシステムを実現することです。一度に 1 台のコンピュータにコンポーネントをインストールし、基本設定を行うため、ほかのコンピュータ上のコンポーネントと正常に相互動作するような設定値をあらかじめ決定しておく必要があります。

相互動作のための設定値は、あるコンポーネントインスタンスが別のコンポーネントインスタンスと通信するために使用する URL やポート番号などの値が含まれます。たとえば、ソリューションで Access Manager を使用する場合、まず Directory Server インスタンスなどの LDAP リポジトリをインストールして設定する必要があります。次に、Access Manager インスタンスをインストールし設定するときに、すでにインストールおよび設定済みの LDAP ディレクトリと相互動作するように Access Manager を設定する値を指定する必要があります。

Java ES インストーラは、ソリューションで使用されているほかのコンピュータにどのようなコンポーネントがインストールされているかについては認識しません。たとえば、Access Manager をインストールするとき、適切な LDAP ディレクトリが位置している場所をインストーラは認識していません。インストールと設定のプロセスを確実に成功させるためには、Access Manager インスタンスおよび Directory Server インスタンス間での正常な相互動作につながるインストールと設定の値を、あらかじめ決定しておく必要があります。これらの値をインストール計画に含めます。そのあと、コンポーネントをインストールし設定するときに、計画に入れた値を入力すれば、互いに相互動作するコンポーネントが正常に設定されます。

図 3–2 に示すような、一連のインストールおよび設定タスクを実行する場合があります。

システム 01:
1. Directory Server をインストールします。ディレクトリサービスポート、ディレクトリサービス管理者のアカウントとパスワード、およびディレクトリルート DN を指定します。

システム 02:
1. Access Server をインストールします。ディレクトリサービス URL (http://system01:389) での Directory Server インスタンス、ディレクトリサービス管理者のアカウントとパスワード、およびディレクトリルート DN を識別します。

図 3–2 コンポーネントの相互動作のための設定

ソリューションのアーキテクチャーがどのようなものであれ、コンポーネントの設定と、正しく相互動作する分散ソリューションの実現のために必要なすべての設定値を含むインストール計画を作成する必要があります。

冗長性戦略

本稼働環境での使用を想定したほとんどのソリューションでは、何らかの種類の冗長化を利用します。冗長性戦略では、コンポーネントの複数のインスタンスを使って 1 つのサービスを提供します。冗長化はサービス品質要件を満たすために利用されます。たとえば、スループットを増強してパフォーマンス要件を満たす目的や、シングルポイント障害を回避して信頼性要件を満たす目的で冗長化を使用します。

Java ES コンポーネントのインスタンス冗長化のために利用できる戦略は 3 つあります。負荷分散、Sun Cluster ソフトウェアによるクラスタリング、および、Directory Server レプリケーションです。ここでは、これらの戦略のそれぞれに対して推奨されるインストールおよび設定手順の概略を示します。

- 負荷分散は、ハードウェアまたはソフトウェアのどちらかで実装できます。負荷分散をもっとも効果的にセットアップするには、負荷分散されるコンポーネントの 1 つのインスタンスをインストールおよび設定したあと、最初のインスタンスによって提供されるサービスがロードバランサを通じて利用可能なことをテストします。サービスが利用可能なことを検証したあと、配備アーキテクチャーで必要されているコンポーネントの追加インスタンスをインストールし、設定します。このように段階化されたインストールと設定のアプローチにより、設定に伴う問題の診断と解決が容易になります。
- クラスタリングは複数の手順で実装されます。最初のステップでは、Sun Cluster ソフトウェアをインストールし、クラスタを確立および設定します。次のステップでは、クラスタ内で実行するコンポーネントをインストールします。たとえば、図 2–1 に示されたクラスタを実装するための最初のステップでは、コンピュータ STR1 および STR2 上に Sun Cluster ソフトウェアをインストールし、クラ

スタを確立および設定します。2 番目のステップでは、Messaging Server と Calendar Server をインストールし、設定します。最後の 3 番目のステップでは、Messaging Server および Calendar Server の Sun Cluster データサービスをインストールし、設定します。それぞれの Sun Cluster データサービスが設定されると、クラスタノードは Messaging Server と Calendar Server のインスタンスを認識します。

- Directory Server レプリケーションも、複数のステップで実装されます。たとえば、マルチマスターレプリケーションを実装する場合、最初のステップは、すべての Directory Server インスタンスのインストール、設定、および検証です。2 番目のステップでは、1 つを除いたすべての Directory Server インスタンスをシャットダウンします。3 番目のステップでは、ソリューション内のほかのコンポーネントをインストールおよび設定します。スキーマまたはディレクトリ構造への変更は、実行中の 1 つの Directory Server インスタンスに対して行われます。最終ステップでは、ソリューション内のすべてのコンポーネントインスタンスがインストール、設定、および検証されたあと、Directory Server のほかのインスタンスを再起動し、レプリケーション機能を使って同期とフェイルオーバーを設定します。これにより、変更および更新されたディレクトリデータがすべての Directory Server インスタンスにコピーされます。

配備アーキテクチャーでこれらの冗長性戦略のいずれかを使用するときは、コンポーネントの複数のインスタンスをインストールし、単独のサービスとして動作するようそれらのインスタンスを設定する手順を、インストール計画に含める必要があります。

LDAP スキーマと LDAP ディレクトリツリーの構造

ほとんどの Java ES ソリューションには Directory Server が含まれます。Directory Server でソリューションをインストールし設定する場合は、ディレクトリスキーマとディレクトリツリー構造の両方を設定する値を入力します。インストール計画では、正しい LDAP スキーマとディレクトリツリー構造を作成する入力値をリストする必要があります。

LDAP スキーマとディレクトリツリー構造は、インストール計画を開始する前に指定します。インストール計画には、インストーラを実行して指定したスキーマとディレクトリ構造を作成するときに入力する値を含めます。スキーマとディレクトリツリー仕様の例は、20 ページの「ユーザー管理仕様の作成」を参照してください。

LDAP スキーマは、次のインストールおよび設定プロセスによって確立されます。

1. Directory Server をインストールすると、スキーマ 1 のディレクトリが自動的に確立されます。スキーマを選択するための入力は不要です。
2. Access Manager をインストールすると、ディレクトリが自動的に変更され、そのディレクトリがスキーマ 2 に変換されます。スキーマを選択するための入力は不要です。

3. Communications Suite コンポーネントが含まれるソリューションでは、Directory Preparation Tool を実行すると、Messaging Server、Calendar Server、および Communications Express で使用するためのスキーマが拡張されます。Directory Preparation Tool は、スキーマ 1 ディレクトリとスキーマ 2 ディレクトリの両方を拡張します。Directory Preparation Tool に対する入力値はインストール計画にリストされます。
4. Communications Suite コンポーネントが含まれるソリューションでは、Delegated Administrator を実行すると、スキーマが拡張され、特定のサービスに対してユーザーを承認および認証するために使われるオブジェクトクラスと属性が追加されます。入力値は、ソリューションが提供するサービスの内容によって異なります。入力値をインストール計画に記述します。

インストールと設定のプロセスでは、基本ディレクトリツリー構造も確立されます。

1. Directory Server をインストールすると、ベースサフィックス (ディレクトリツリーのルート) が作成されます。ベースサフィックスは、Java ES インストーラが Directory Server をインストールするときに必要な入力値です。インストール計画では、インストールプロセスのための入力値の 1 つとしてベースサフィックスを指定します。
2. Messaging Server をインストールして設定すると、ディレクトリツリーが分岐して LDAP 組織が作成されます。この組織は、Messaging Server インスタンスによって管理される電子メールドメインを表します。組織の名前は、Messaging Server の設定ウィザードで必須の入力です。インストール計画では、Messaging Server の設定プロセス用の入力値の 1 つとして組織 DN を指定します。
3. Calendar Server、Communications Express、Delegated Administrator、および Instant Messaging のインストールと設定では、これらのコンポーネントがユーザーデータを検索するディレクトリ構造内の場所を指定します。LDAP DN は各コンポーネントの設定ウィザードの必須入力であり、インストール計画では各設定ウィザード用の入力値として DN を指定します。ソリューションで Access Manager シングルサインオンを使用する場合、これらのコンポーネントがすべて同じ場所 (Messaging Server の設定ウィザードで作成された組織) にユーザーデータを保存するように設定する必要があります。これら全コンポーネントの設定ウィザードで、同じ LDAP DN が入力されます。インストール計画では、すべての設定ウィザードに対する入力値の 1 つとして組織 DN を指定します。

LDAP ベースサフィックスおよび電子メール組織の名前を、ユーザー管理仕様をもとに決定し、インストール計画に追加します。ユーザー管理仕様の詳細については、20 ページの「ユーザー管理仕様の作成」を参照してください。

Java ES インストーラの動作

ここでは、インストール計画に影響する Java ES インストーラの一部の動作について説明します。

インストーラはローカルである

Java ES インストーラは、コンポーネントソフトウェアを一度に 1 台のコンピュータにインストールします。ほとんどのソリューションは分散構成であり、インストーラを複数回実行する必要があります。インストール計画には、インストールの実行のたびに行う手順を含める必要があります。ここでは、アーキテクチャーを実装するためにインストーラを実行する必要がある回数を、配備アーキテクチャーの分析に基づいて決定する方法について説明します。

ソリューションの中には 1 台のコンピュータにのみインストールされるものがあり、そのようなソリューションのインストール計画は、インストーラを 1 回だけ実行するための手順を定義します。インストーラの実行が 1 回で済むのは次のようなソリューションです。

- Java ES の機能を評価する目的で、多数のコンポーネントを 1 台のコンピュータにインストールする。
- 確立済みのソリューションに 1 つのコンポーネントインスタンスを追加する。これには、既存のコンポーネントに対する依存性を持つコンポーネントインスタンスの追加が含まれます。

ほとんどのソリューションは、複数のコンピュータに分散されます。そのようなソリューションのインストール計画では、ソリューション全体をインストールおよび設定するために、複数回のインストーラ実行を記述する必要があります。そのようなソリューションを分析するには、次のガイドラインを使用します。

- ほとんどの場合、1 台のコンピュータ上で複数のコンポーネントを組み合わせるときは、インストーラを 1 回だけ実行します。これは特に、インストーラが「今すぐ設定」モードで動作する場合に当てはまります。「今すぐ設定」モードでは、インストーラは Web コンテナと、そのコンテナ内で実行されるコンポーネントの両方をインストールできるからです。これらの場合、インストール計画では、そのコンピュータ上での 1 回のインストーラ実行と、そのコンピュータに対して指定されたすべてのコンポーネントの選択を記述します。

ヒント – 一部のコンポーネントは、「今すぐ設定」モードでもインストーラによる設定はできません。そのようなコンポーネントをコンピュータにインストールするとき、設定プロセスはそれぞれのコンポーネントに対して設定ウィザードを実行することによって完了します。それらのコンポーネントを、インストーラによって設定されるコンポーネントと組み合わせてインストールする場合には、まずインストーラを実行します。インストーラの実行後、インストーラによって設定されないコンポーネントに対して設定ウィザードを実行することによってプロセスを完了します。これらの場合、インストール計画では、インストーラの実行と、複数の設定ウィザード実行の正しいシーケンスを記述する必要があります。

- コンポーネントの組み合わせによっては、1 台のコンピュータ上でインストーラを複数回実行しないとインストールできない場合があります。そのような組み合わせには、次のものがあります。
  - Web コンテナを含む一部のコンポーネントの組み合わせ。Web Server または Application Server を「あとで設定」モードでインストールする場合、Web コンテナで実行するほかのコンポーネントをインストールするには、あらかじめ Web Server または Application Server のインスタンスを設定する必要があります。ソリューションでサードパーティー製の Web コンテナを使用する場合、Web ベースの Java ES コンポーネントをインストールする前に、Web コンテナをインストールし、起動し、検証する必要があります。インストール計画には、各コンピュータ上での複数回のインストーラ実行の手順を記述する必要があります。
  - Sun Cluster ソフトウェアを使用するコンポーネントの組み合わせ。クラスタにインストールされるコンポーネントがクラスタファイルシステム上にインストールされる場合、先に Sun Cluster ソフトウェアをインストールし、クラスタファイルシステムを作成してからでないと、クラスタノードにほかのコンポーネントをインストールできません。インストール計画には、各コンピュータ上での複数回のインストーラ実行の手順を記述する必要があります。

この節の目的は、インストール計画では時として、1 台のコンピュータ上でのインストーラと設定ウィザードの実行、または、1 台のコンピュータ上での複数回のインストーラ実行を記述しなければならないという概念を紹介することです。さまざまなコンポーネントの組み合わせに対する実際のインストール手順の詳細については、40 ページの「インストール計画の作成」を参照してください。

インストーラの動作モード

インストーラは、「今すぐ設定」と「あとで設定」の 2 つの異なるモードで動作します。2 つのモードの違いは次の点です。

- 「今すぐ設定」モードでは、インストーラは全部ではなく一部のコンポーネントの実行可能インスタンスを設定します。「今すぐ設定」モードで設定されるコンポーネントは、インストーラが完了した時点で起動と検証が可能になります。インストーラの実行後、残りのコンポーネントの実行可能インスタンスが、コンポーネントの設定ウィザードを実行することによって作成されます。インストーラによって設定されるコンポーネントについては、インストール計画にインストーラの実行時に入力する設定値を記述する必要があります。インストーラの実行後に設定されるコンポーネントについては、インストール計画に、設定ウィザードの実行手順と、設定ウィザードの実行時に入力する設定値を記述する必要があります。

ヒント – 「今すぐ設定」モードの重要な特長の 1 つは、Web コンテナと、その Web コンテナ内で動作するコンポーネントを同時にインストールできることです。コンポーネントはインストーラによって自動的に Web コンテナに配備されます。

- 「あとで設定」モードでは、インストーラはコンポーネントソフトウェアのファイルをコンピュータにコピーしますが、実行可能なインスタンスは作成しません。インスタンスは、インストーラの実行後に、コンポーネント設定ウィザードを実行することによって作成します。インストール計画には、設定ウィザードの実行手順と、設定ウィザードの実行時に入力する設定値を記述する必要があります。

選択した設定オプションは、インストールセッション全体に適用されます。「今すぐ設定」モードと「あとで設定」モードで一部のコンポーネントをコンピュータ上にインストールしたい場合は、インストーラを複数回実行する必要があります。

インストーラの互換性チェック

Java ES インストーラでは、依存性と互換性に関するいくつかのチェックが実行されます。ただし、インストーラでチェックできるのはローカルコンピュータのみです。たとえば、分散ソリューションで Access Manager をインストールする場合、インストーラでは、インストールする Access Manager との互換性がリモート Directory Server にあるかどうかのチェックを行うことはできません。

同じ Java ES リリースのすべてのコンポーネントを含む完全に新規のソリューションをインストールし設定する場合、互換性が問題になることは考えられません。確立済みのソリューションに新しいコンポーネントを追加している場合や、既存のコンポーネントの周囲に Java ES ソリューションを構築している場合、そのことが問題となる可能性があります。たとえば、すでに Directory Server を使用しており、Access Manager と Portal Server を使用するソリューションを既存の Directory Server の周囲に構築している場合、コンポーネント間の互換性が問題となります。新しいコンポーネントのインストールと設定を開始する前に、コンポーネントの互換性を確認する必要があります。

- コンポーネントの依存性チェック。Java ES インストーラは、ローカルホスト上でのみ、インストールするように選択したほかのコンポーネントに必要なコンポーネントを除外しないようにします。分散ソリューションでは、インストーラは、リモートコンポーネントの存在を検証するためのリモートホストのチェックは行いません。この場合、リモートコンポーネントに互換性があり、適切な稼働状態にあることを確認する必要があります。
- アップグレード。Java ES インストーラは、インストールされている Application Server、Message Queue、HADB、および Java DB と、インストールするコンポーネントとの互換性をチェックし、インストール時にコンポーネントをアップグレードするかどうかをユーザーに問い合わせます。

Java ES インストーラは、共有コンポーネントのアップグレードを行います。このトピックの詳細については、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「既存ホストの調査」を参照してください。

インストールに関するその他の問題

この節では、一部のソリューションで発生するいくつかの固有の問題を、詳細な情報の参照先とともに示します。

表 3–2 考慮する必要のあるインストールの問題

ソリューションに必要なもの	ガイドラインまたは取扱説明書
Solaris 10 ゾーンの使用	Solaris 10 ゾーンへのインストールを予定している場合、付録 A を参照してください。
Directory Server 暗号化の使用	Directory Server インスタンス上で LDAPS (SSL over LDAP) を設定します。
Access Manager でのサードパーティー製 Web コンテナの使用	サードパーティー製 Web コンテナ (BEA WebLogic Server または IBM WebSphere Application Server) を Portal Server および Access Manager と組み合わせて使用できます。これらのコンテナは、コンテナに依存する Java ES コンポーネントをインストールする前にインストールして実行する必要があります。 Access Manager SDK に対してサードパーティー製 Web コンテナを使用するには、インストール後に Access Manager SDK を手動で設定する必要があります。『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「コンテナの設定を使用する Access Manager SDK の例」を参照してください。 注意: Portal Server は、Solaris OS 上ではサードパーティー製 Web コンテナのみを使用できます。 注意: Access Manager と Portal Server では同じ種類の Web コンテナを使用することが推奨されます。
負荷分散プラグインに対する Apache Web Server の使用	Apache Web Server は、Application Server の負荷分散プラグインで使用することができます。この場合、Apache Web Server は、このサーバーに依存する Java ES コンポーネントをインストールする前にインストールし、実行する必要があります。
スキーマ 1 LDAP の使用	スキーマ 1 配備については、Access Manager を使用できません。
シングルユーザーエントリとシングルサインオンの設定	シングルサインオンを行うには Access Manager が必要です。
HADB を使用した高可用性の設定	高可用性のための HADB の設定手順の概要は、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「Web とアプリケーションサービスの例」に記載されています。

表 3–2 考慮する必要のあるインストールの問題 (続き)

ソリューションに必要なもの	ガイドラインまたは取扱説明書
Application Server 負荷分散	Application Server の負荷分散プラグインの使用手順の概要は、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「Web とアプリケーションサービスの例」に掲載されています。
非ルート所有権	Application Server または Web Server に対して非ルート所有権が必要な場合は、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の「root 以外のユーザーでのインストール例」を参照してください。

インストール計画の作成

配備アーキテクチャーと実装仕様では、ソリューションの最終状態を記述します。配備アーキテクチャーでは、何個のコンポーネントインスタンスがインストールされるか、コンポーネントインスタンスがどのコンピュータシステムにインストールされるか、および、コンポーネントインスタンスがどのように相互動作するかを示します。配備アーキテクチャーで記述された状態に到達するために、ソリューション全体のインストールと設定が完了するまで、ソリューション内のコンピュータシステムの 1 台ずつにコンポーネントインスタンスをインストールして設定する必要があります。インストール計画では、ソリューション内のすべてのコンポーネントインスタンスを正しい順序でインストールして設定するための手順を定義する必要があります。

インストールおよび設定計画を作成するには、コンポーネントの依存性やそのほかのインストールの問題に関する知識を、Java ES の配備アーキテクチャーおよび実装仕様に応用する必要があります。ソリューション内のコンポーネントインスタンスをインストールして設定する正しいシーケンスと、コンポーネントインスタンスの相互動作を実現する正しい設定入力値を決定する必要があります。

この節では、配備アーキテクチャーと一連の仕様を分析して、インストール計画を作成する方法について説明します。通常は、次のような手順で作業を開始します。

1. 計画を記録するためのテキストファイル、空白の紙、またはそのほかのメディアを開きます。
2. 配備アーキテクチャーで、各コンピュータシステム上のコンポーネントを検証し、どのようなコンポーネント依存性が存在するかを判断します。
3. ほかのコンポーネントへの依存性を持たないコンポーネントインスタンスを識別します。そのようなインスタンスは通常、Directory Server のインスタンスです。インストール計画の作成は、指定されたコンピュータシステムにそのようなコンポーネントインスタンスをインストールするための手順の定義から始めます。そのようなコンピュータシステムと、それらのシステムにインストールされるコンポーネントインスタンスを記録して、インストール計画を開始します。

4. それらのコンピュータシステム上のコンポーネントインスタンスについて、対象のソリューションにおける正しいインストール/設定値を決定します。これらの設定値をインストール計画に追加します。
5. 残りのコンポーネントの中で、Directory Server にのみ依存性を持つコンポーネントを特定します。それらは通常、Access Manager が動作するコンピュータシステムです。次に、それらのコンピュータシステムをインストール計画にリストします。
6. コンポーネントの依存性の順序で仕様の分析を続けます。必要な設定値を決定し、それらのコンポーネントインスタンスを計画に記録します。

たとえば、このプロセスを使用して、図 2–1 に示した配備アーキテクチャーを分析する場合、表 3–3 のようなインストール計画を作成します。

表 3–3 は、インストール計画の最初の 8 つのステップを示したものです。この計画の構造を明確にするために、個別の設定値についてはリストしていません。この計画では、次のことに注意します。

- 計画では、ソリューション内のコンピュータを、コンポーネントインスタンスがインストールおよび設定される順序に従ってリストします。
- インストールのシーケンスは、ソリューションレベルの依存性とローカル依存性の両方を適用することによって決定されます。ソリューションレベルの依存性の適用により、Directory Server、Access Manager、Messaging Server、そして Calendar Server という基本シーケンスが決定されます。このシーケンスにローカルの Communications Express 依存性を適用することにより、コンピュータ AM1 および AM2 上に Web Server のインスタンスが、コンピュータ mscs01 および mscs02 上に Sun Cluster ソフトウェアおよび Sun Cluster エージェントが追加されます。
- 計画には、Java ES ソリューションで採用されるすべての冗長性戦略に対応する、概略のインストール手順と設定手順が含まれます。DS1 および DS2 に対するタスクのリストは、Directory Server マルチマスターレプリケーションに対する計画の例です。AM1 および AM2 に対するタスクのリストは、負荷分散されるコンポーネントに対する計画の例です。STR1 および STR2 に対するタスクのリストは、Sun Cluster 設定で動作するコンポーネントに対する計画の例です。
- STR1 および STR2 に対するタスクは、1 台のコンピュータに複数のコンポーネントをインストールして設定する例を提供します。最初のインストーラ実行で、Sun Cluster コアコンポーネントがインストールされます。Sun Cluster コアコンポーネントを設定したあと、再度インストーラを実行して Messaging Server と Calendar Server をインストールします。これらのコンポーネントは、それぞれの依存性に従って順番に設定されます。コンピュータ上での 3 回目のインストーラ実行では、Messaging Server と Calendar Server の存在に応じて、Sun Cluster Agents for Messaging Server および Sun Cluster Agents for Calendar Server がインストールされます。

表 3–3 サンプル配備アーキテクチャーのインストール計画要約

コンピュータ	インストールと設定のタスク
DS1	1. このコンピュータ上で Java ES インストーラを実行します。ユーザー管理仕様で指定された設定値を使用して、Directory Server インスタンスをインストールおよび設定します。 2. Directory Server インスタンスを起動し、検証します。
DS2	1. このコンピュータ上で Java ES インストーラを実行します。ユーザー管理仕様で指定された設定値を使用して、Directory Server インスタンスをインストールおよび設定します。 2. Directory Server インスタンスを起動し、検証します。 3. 両方の Directory Server インスタンスに対してロードバランサが正しく機能していることを検証します。 4. DS2 内の Directory Server インスタンスを停止します。DS1 上の Directory Server インスタンスはそのまま実行させておきます。
AM1	1. このコンピュータ上で Java ES インストーラを実行します。Access Manager インスタンスをインストールおよび設定します。Access Manager インスタンスは、負荷分散される Directory Server インスタンスによって作成される論理ディレクトリサービスと相互動作するように設定します。 2. Access Manager インスタンスを起動し、検証します。 3. 負荷分散のために Access Manager インスタンスを設定します。
AM2	1. このコンピュータ上で Java ES インストーラを実行します。Access Manager インスタンスをインストールおよび設定します。Access Manager インスタンスは、負荷分散される Directory Server インスタンスによって作成される論理ディレクトリサービスと相互動作するように設定します。 2. Access Manager インスタンスを起動し、検証します。 3. 負荷分散のために Access Manager インスタンスを設定します。 4. Access Manager コンソールを使用して、Access Manager に対するディレクトリエントリを変更します。 5. 2 つの Access Manager インスタンスが、処理が負荷分散されて正しく機能していることを検証します。

表 3–3 サンプル配備アーキテクチャーのインストール計画要約 (続き)

コンピュータ	インストールと設定のタスク
STR1	1. Java ES インストーラを実行します。Sun Cluster コアコンポーネントをインストールします。 2. Sun Cluster 設定用にコンピュータを準備します。このステップには、Sun Cluster ソフトウェアによって使用されるファイルシステムの作成とマウントが含まれます。 3. Sun Cluster の設定ウィザードを実行します。クラスタを確立および設定します。
STR2	1. Java ES インストーラを実行します。Sun Cluster コアコンポーネントをインストールします。 2. Sun Cluster 設定用にコンピュータを準備します。このステップには、Sun Cluster ソフトウェアによって使用されるファイルシステムの作成とマウントが含まれます。 3. Sun Cluster の設定ウィザードを実行します。クラスタを確立および設定します。 4. STR1 および STR2 上で NTP (Network Timing Protocol) の設定を完了します。 5. quorom デバイスを (両方のコンピュータに接続された) クラスタに追加します。 6. クラスタファイルシステムとリソースグループを作成し、仮想ホスト名と IP アドレスを設定します。 7. クラスタのフェイルオーバー機能を検証します。

表 3–3 サンプル配備アーキテクチャーのインストール計画要約 (続き)

コンピュータ	インストールと設定のタスク
STR1	1. Java ES インストーラを実行します。Messaging Server および Calendar Server をインストールします。 2. コンピュータ DS1 上で Directory Server Preparation Tool を実行します。 3. Messaging Server の設定ウィザードを実行して、Messaging Server のインスタンスを作成します。ユーザー管理仕様に従って LDAP ディレクトリツリー内にブランチを作成する設定値を指定します。負荷分散された Access Manager インスタンスおよび負荷分散された Directory Server インスタンスと相互動作するように Messaging Server インスタンスを設定する設定値を指定します。 4. シングルサインオン用に Messaging Server を設定します。 5. Messaging Server インスタンスを起動し、検証します。 6. Calendar Server の設定ウィザードを実行して、Calendar Server のインスタンスを作成します。Messaging Server の設定によってユーザーおよびグループデータ用に作成された LDAP ブランチを使用するようにインスタンスを設定する設定値を指定します。負荷分散された Access Manager インスタンスおよび負荷分散された Directory Server インスタンスと相互動作するように Calendar Server インスタンスを設定する設定値を指定します。 7. コンピュータ STR2 上で、Calendar Server のユーザー、ユーザーグループ、およびディレクトリを作成します。 8. Calendar Server の設定ファイルを編集します。コンピュータの IP アドレスの代わりに仮想 IP アドレスを使用するように設定パラメータを設定します。 9. シングルサインオン用に Calendar Server を設定します。 10. Calendar Server インスタンスを起動し、検証します。
STR1	1. Java ES インストーラを実行します。Sun Cluster Agent for Messaging Server および Sun Cluster Agent for Calendar Server をインストールします。 2. Messaging Server エージェントを使用して、Messaging Server リソースを作成および有効化します。 3. Messaging Server リソースの、STR1 から STR2 へのフェイルオーバーを検証します。 4. Calendar Server エージェントを使用して、Calendar Server リソースを作成および有効化します。 5. Calendar Server リソースの、STR1 から STR2 へのフェイルオーバーを検証します。
STR2	mscs01 上で設定したインスタンスは、共有インスタンスとして自動的に認識されます。

◆ ◆ ◆ 付 録 A

Java ES と Solaris 10 ゾーン

この付録では、Solaris 10 ゾーンに Java ES コンポーネントをインストールおよび設定するときに起きる問題について説明し、それらの問題に対処するために推奨されているいくつかの手法を示します。この付録は、次の節で構成されています。

- 45 ページの「ゾーンとは何か」
- 48 ページの「Java ES にゾーンを使用する理由」
- 49 ページの「Java ES コンポーネントのゾーン制限」
- 52 ページの「Java ES インストーラでのゾーンサポート」
- 57 ページの「Java ES で推奨されるゾーンの使用法」
- 61 ページの「特殊なケースまたは例外」
- 62 ページの「サンプル: 疎ルートゾーンに Application Server をインストールする」

ゾーンとは何か

ゾーンは、Solaris 10 オペレーティングシステムのアプリケーションおよびリソース管理機能です。この機能により、オペレーティングシステムはアプリケーションに対して、分離およびセキュリティー保護された複数の仮想オペレーティングシステム環境 (ゾーン) として表されます。これらのゾーンを使用すると、オペレーティングシステムに独立性を持たせながら、ある程度の集中リソース管理が可能になるという利点があります。そのため、アプリケーションを互いに別々のゾーンにインストールおよび実行して分離しながらも、特定のオペレーティングシステムリソースを中央に割り当てて集中管理することが可能です。

複数のゾーンをサポートするオペレーティングシステムの観点から見れば、オペレーティングシステムリソースにはプロセス管理、メモリー、ネットワーク構成、ファイルシステム、パッケージレジストリ、ユーザーアカウント、共用ライブラリ、および場合によってはインストール済みのアプリケーションが含まれます。

複数ゾーン環境の構造

複数ゾーン環境は、1 つの大域ゾーン (デフォルトのオペレーティングシステム) と 1 つ以上の非大域ゾーンで構成されます。大域ゾーンには、大域 (ゾーン) 管理者が複数の非大域ゾーンに割り振ることのできるリソースが含まれます。非大域ゾーンには次の機能があります。

- セキュリティー。分散サービスを非大域ゾーンで実行すると、セキュリティー違反が発生した場合の損害を抑えることができます。あるゾーンでソフトウェアのセキュリティーの欠陥を突いて侵入に成功したとしても、侵入はそのゾーンに限られます。非大域ゾーン内で使用できる特権は、大域ゾーンで使用できる特権の一部のみです。
- ランタイムの分離。複数のアプリケーションに異なるレベルのセキュリティー、大域リソースへの排他アクセス、または個別の設定が必要となる場合でも、非大域ゾーンを使用することで、それらのアプリケーションを同じコンピュータ上に配備することが可能となります。たとえば、異なるゾーンで実行中の複数のアプリケーションは、各非大域ゾーンに関連付けられている別個の IP アドレスを使用することで、同じネットワークポートにバインドできます。これにより、アプリケーションが互いのネットワークトラフィック、ファイルシステムデータ、プロセスの活動などを監視したり妨害したりすることが防止されます。
- 管理の独立。オペレーティングシステム環境を仮想化することで、非大域ゾーンをそれぞれ個別に管理できます。大域管理者ではなくゾーン管理者が非大域ゾーンで行う、ユーザーアカウントの作成、ソフトウェアのインストールと設定、プロセスの管理などのアクションは他のゾーンに影響しません。

非大域ゾーンには、完全ルートゾーンと疎ルートゾーンの 2 種類があります。

- 完全ルートゾーン: 大域ゾーン上に存在するファイルシステムの読み取り/書き込みコピーがあります。完全ルートゾーンが作成されると、大域ゾーン上にインストールされているすべてのパッケージは完全ルートゾーンで使用可能になります。パッケージデータベースを作成すると、ゾーンを専用かつ単独で使用するために、すべてのファイルが完全ルートゾーンにコピーされます。
- 疎ルートゾーン: 大域ゾーンに存在するファイルシステムの一部の読み取り/書き込みコピーのみがあり (そのために疎ルートという名前で呼ばれる)、それ以外のファイルシステムは大域ゾーンから読み取り専用でループバック仮想ファイルシステムとしてマウントされます。大域管理者は、疎ルートゾーンの作成時に、この疎ルートゾーンとどのファイルシステムを共有するかを選択します。デフォルトでは、`/usr`、`/lib`、`/sbin`、および `/platform` が読み取り専用ファイルシステムとして共有されます。大域ゾーンにインストールされているすべてのパッケージが、疎ルートゾーンで利用できるようになります。パッケージデータベースが作成され、マウントされているファイルシステム内のすべてのファイルがゾーンで共有されます。

完全ルートゾーンと疎ルートゾーンのどちらを選択するか

完全ルート非大域ゾーンと疎ルート非大域ゾーンのどちらを使用するかは、リソースの効率化と管理統制のどちらに重点を置くかによって左右されます。完全ルートゾーンではメモリーや他のリソースを犠牲にして管理統制 (独立性や分離) を最大化できますが、疎ルートゾーンでは管理の独立性を犠牲にして実行可能ファイルや共用ライブラリが効率的に共用されるよう最適化されます (ディスク使用量はずっと少なくなる)。疎ルートゾーンが完全ルートゾーンよりどれだけパフォーマンスに優れているかを測る尺度は今のところありません。通常、それはソフトウェアによって異なります。

パッケージの伝播

パッケージが大域ゾーンにインストールされると、そのパッケージはすべての非大域ゾーンでデフォルトで使用可能となります。このプロセスをパッケージの伝播といいます。この伝播が実行されるには、新たに作成された非大域ゾーンを完全にブートする、つまり実行状態にする必要があります。伝播によって、大域ゾーンにインストールされているパッケージが、ローカルつまり非大域ゾーンから表示および使用できるようになります。伝播によってアプリケーションパッケージのライフサイクル管理 (インストール、アップグレード、アンインストール) は大域管理者が一括して行えるようになりますが、アプリケーション設定とランタイム管理は (非大域) ゾーン管理者が行います。

完全ルートゾーンの場合、伝播は、大域ゾーンから完全ルートゾーンへのインストール済みファイルの自動コピーと、レジストリ情報の自動同期化によって行われます。疎ルートゾーンの場合、伝播は大域ルートゾーンと疎ルートゾーンとが共有する読み取り専用ファイルにより、レジストリ情報の自動同期化によって行われます。

非大域ゾーンへのパッケージの伝播は、内部パッケージ属性を使用してパッケージレベルで制御されます。これらの属性の値の一部、少なくともデフォルト値については、インストール時に `pkgadd –G` オプションを使用して属性値を上書きすることで、伝播を無効にすることができます。一度インストールした後は、パッケージをアンインストールしてインストールし直さないかぎり、パッケージの伝播の動作を変更できません。たとえば、パッチでパッケージの伝播の動作を変更することはできず、アップグレードする対象のパッケージの伝播の動作に合わせてパッチを適用する必要があります。

Java ES にゾーンを使用する理由

異なるゾーンで実行するアプリケーションどうしが互いに孤立している状態は、異なるコンピュータの複数のオペレーティングシステムでアプリケーションを実行するときの状態と似ています。ですから、異なるコンピュータ上で Java ES コンポーネントをインストール、設定、および実行することでそれぞれのコンポーネントを孤立させてセキュリティー保護する代わりに、同じコンピュータの中の異なるゾーンにそれらのコンポーネントをインストールし、設定および実行することができます。

そのようにして Java ES コンポーネントを統合することにより、リソースをより効率的に活用できるようにもなります。複数台の専用コンピュータを中途半端に使って実行していた Java ES コンポーネントを、1 台のコンピュータの複数の非大域ゾーンで実行することが可能になるからです。大域管理者は、ゾーンで実行するコンポーネントのリソース要件に応じて、異なるゾーンの間でリソースを動的に割り当てることができます。このことを実行するには、各種コンポーネントのリソース要件に関して詳細な知識と理解が必要になる場合があります。

複数ゾーン環境は、他の目的を達成する面でも役立ちます。

- バージョンの分離: バージョンが異なる Java ES コンポーネントを、異なるゾーンで並行して実行させることができます。この方法で、時間をかけて Java ES のバージョンを変更できます。たとえば、Java ES Release 4 コンポーネントと Java ES Release 5 コンポーネントを別々の非大域ゾーンで並行して実行できます。このような形でバージョンの分離を行えるようにするために、ライフサイクル管理は、設定およびランタイム管理と同様、ゾーン管理者に委任されています。
- ライフサイクル管理の集約: Java ES の制限により完全にはサポートされていませんが、ゾーンは Java ES コンポーネントのライフサイクル管理の集約を可能にします。コンポーネントのインストール、アップグレード、およびアンインストールは大域ゾーンで可能ですが、実行時の分離、セキュリティー、スケーラビリティー、およびその他の必要に備えて、設定と実行は複数の非大域ゾーンで行います。ライフサイクル管理の集約は、特定のコンポーネントのインスタンスがいくつも異なるゾーンで実行されている場合や、そのようなインスタンスが確実に同一のリリースバージョンに同期されるようにする場合に便利です。

  たとえば、Application Server を大域ゾーンにインストールしておいて、複数のインスタンスをいくつかの非大域ゾーンで実行することもできます。さまざまな Application Server インスタンスが Access Manager、Portal Server、またはその他の Java ES コンポーネントをサポートできます。これらは、異なる非大域ゾーンの同一のコンポーネントでも異なるコンポーネントでもかまいません。あるいは、別々の開発チームが別々の Application Server インスタンスを異なるゾーンで使用するということも可能です。

  このことを実現するために、ライフサイクル管理は大域管理者が行い、設定とランタイム管理はそれぞれのゾーン管理者に委任されています。この方法には、アップグレードなどのライフサイクル管理タスクを実行するときに、広範な調整が必要となります。

- 組織上の独立性: すべてが同じコンピュータ上で共存し実行する Java ES コンポーネントの配備またはそのランタイムインスタンスを、組織別に別個に維持管理することができます。たとえば、異なる開発者グループが Java ES コンポーネントの独自のインスタンスを使用したり、組織ごとにテスト、試作段階、本稼働用で異なる Java ES 配備を使用したりできます。組織上の独立性は様々な方法で達成できますが、その方法は Java ES のライフサイクル管理は集約しつつ設定とランタイム管理はゾーン管理者に委任するか、またはライフサイクル、設定およびランタイムのすべての管理機能をゾーン管理者に委任するかのどちらを目的とするかによって左右されます。

複数ゾーン環境で Java ES を使用して実現する目的、およびその目的に伴う使用のシナリオが異なれば、複数ゾーン環境への Java ES コンポーネントの配備と管理にも異なる方策が必要となります。複数のゾーンの独立性を利用して種々の Java ES コンポーネントとそのランタイムインスタンスを個別に管理するという目的があれば、一方では大域ゾーンの伝播機能を利用して Java ES コンポーネントのライフサイクル管理を簡素化するという目的もあります。

複数ゾーン環境で Java ES を使用するためのインストールおよび管理の方策については、Java ES ソフトウェアの性質ゆえに課せられる複数ゾーン環境の制約についていくらか説明した後で再検討します。

Java ES コンポーネントのゾーン制限

『Sun Java Enterprise System 5 技術の概要』で説明されているように、Java ES コンポーネントはさまざまなタイプにグループ分けされています。それによると、主な Java ES インフラストラクチャーサービスはシステムサービスコンポーネントによって提供されており、システムサービスはサービス品質コンポーネントによって機能が強化されています。この 2 種類の Java ES コンポーネントをここでは製品コンポーネント (Java ES インストーラ内で選択可能なコンポーネント) と呼ぶことにします。

どちらの製品コンポーネントも、Java ES 共有コンポーネントというローカルに共用される 1 つ以上のライブラリに依存します。共有コンポーネントは、Java ES インストーラで製品コンポーネントのインストール時に、インストールされる製品コンポーネントに応じて自動的にインストールされます。Java ES 製品コンポーネントの配備中に、それらが個別に選択されてインストールされたり、設定されることはありません。

Java ES 共有コンポーネントとゾーン

48 ページの「Java ES にゾーンを使用する理由」では、Java ES 製品コンポーネントによるゾーンの使用について論じました。製品コンポーネントは Java ES インストーラで明示的に選択することができ、それぞれ異なるゾーンにインストールして設定することで、希望する配備アーキテクチャーと機能を実現することができます。しか

し、製品コンポーネントが依存する共有コンポーネントを使用することにより、Java ES を複数ゾーン環境に配備する方法にいくらかの制限が生じます。Java ES 共有コンポーネントとゾーンに関して、次の 2 つの問題があります。

共有コンポーネントの同期

Java ES 共有コンポーネントと Java ES 製品コンポーネントとの間には複雑な対話が 30 ほどもあり、それらをテストおよびサポートすることは難しいため、単一のオペレーティングシステムインスタンス内のすべての共有コンポーネントは必ず同じ Java ES バージョンに同期されなければなりません。言い換えると、非ゾーン環境にインストールされている、または Solaris 10 環境内の特定のゾーンにインストールされている、すべての Java ES 共有コンポーネントは同じバージョンでなければなりません。この要件によって、複数ゾーン環境で Java ES を使用できる方法に一定の制限が加えられます。

したがって、この同期要件は次のことをも意味します。

- バージョンの異なる Java ES 共有コンポーネントは、別々のゾーンにしか配置できません。たとえば、Java ES Release 4 共有コンポーネントと Java ES Release 5 共有コンポーネントをそれぞれ別のゾーンにインストールすることはできますが、それらを組み合わせて 1 つのゾーンにインストールすることはできません。
- あるゾーンのいずれかの共有コンポーネントをアップグレードするか、上位バージョンの新しい共有コンポーネントを導入する場合は、そのゾーンのすべての共有コンポーネントも同時にアップグレードする必要があります。共有コンポーネントは下位互換性が保たれている必要があるので、Release 4 製品コンポーネントが Release 5 共有コンポーネントと連動する上での問題はありません。たとえば、Release 5 製品コンポーネントが、1 つ以上の Release 4 製品コンポーネントが配置されているあるゾーンにインストールされているとします。Release 5 製品コンポーネントはいくつかの Release 5 共有コンポーネントを必要とするので、この場合の同期要件は、このゾーンに配置されているすべての Release 4 共有コンポーネントは Release 5 製品コンポーネントのインストールと同時に Release 5 にアップグレードする必要があるということになります。インストール中の Release 5 製品コンポーネントがゾーンにすでにインストールされているものとは異なる共有コンポーネントを必要とする場合であっても、この要件が適用されます。
- 共有コンポーネントが大域ゾーンにインストールされそこから伝播されるようになったなら (伝播ポリシーについては52 ページの「Java ES 伝播ポリシー」を参照)、すべてのゾーンで共有コンポーネントの同期が保たれるよう細心の注意を払う必要があります。このことを怠ると、非大域ゾーンにある旧バージョンの共有コンポーネントと大域ゾーンから伝播された Release 5 共有コンポーネントとが混在しかねません。通常、「細心の注意を払う」とは、共有コンポーネントのライフサイクル管理が大域ゾーン内でのみ行われるようにすることを意味します。詳細については、表 A–2、および 62 ページの「共有コンポーネントの特殊ケース」を参照してください。

共有コンポーネントの同期要件によって、Java ES インストーラが複数ゾーン環境で実行すべき内容に制限が課せられ (詳細は52 ページの「Java ES インストーラでのゾーンサポート」を参照)、複数ゾーン環境で Java ES 製品コンポーネントをインストールおよびアップグレードする手順に影響が及びます。

共有コンポーネントと疎ルートゾーン

複数ゾーン環境での Java ES の使用に影響するもう一つの問題は、疎ルートゾーンの読み取り専用ファイルシステムのために、多数の共有コンポーネントを疎ルートゾーンにインストールできないということです。そのため、ベースディレクトリが `/usr` (デフォルトで大域ゾーンによって共有されるディレクトリ) である共有コンポーネントを大域ゾーンにインストールして、疎ルートゾーンで使用できるようにする必要があります。

多数の Java ES 共有コンポーネントを疎ルートゾーンにインストールできないということは、それらの共有コンポーネントへの依存関係がある製品コンポーネントを疎ルートゾーンに正常にインストールする場合、まず共有コンポーネントを大域ゾーンにインストールして非大域ゾーンに伝播させなければならないということを意味します。

Java ES 製品コンポーネントとゾーン

48 ページの「Java ES にゾーンを使用する理由」で論じられた複数ゾーン環境での Java ES の使用の目的、およびそれに伴う使用のシナリオは、大域ゾーンの伝播機能を活用して Java ES 製品コンポーネントのライフサイクル管理を簡素化するものでした。たとえば、そのような使用のシナリオでは、Java ES 製品コンポーネントのライフサイクル管理を大域管理者が大域ゾーンで実行し、それらのコンポーネントの設定およびランタイム管理はゾーン管理者が非大域ゾーンで実行することが必要になります。

言い換えると、製品コンポーネントは大域ゾーンでインストールおよびアップグレードし、インスタンスは非大域ゾーンで設定および実行するということです。この使用シナリオは、ライフサイクル集中管理の利点と非大域ゾーンが提供する孤立化とセキュリティーとを兼ね備えています。

しかし、このシナリオは各製品コンポーネントを大域ゾーンにインストールし、非大域ゾーンで設定および実行できることを前提としています。このように処理を分けられるかどうかは、各製品コンポーネントの設定方法、設定と動的アプリケーションデータの格納場所、バイナリを実行して設定データを検出する方法、およびアップグレードの実行方法に依存します。たとえば、インストールまたはアップグレードの前後にスクリプトが何を実行するかによって左右される場合があります。スクリプトがコンポーネントインスタンスを開始または停止するか、設定データへのリンクを作成するか、それ以外にライフサイクル管理と設定管理との区別を付きにくくさせるタスクを実行するかというようなことです。

また、完全ルートゾーンと疎ルートゾーンのどちらで設定が実行されるかによって左右される場合もあります。たとえば、製品コンポーネントの設定スクリプトが読み取り専用ファイルシステムを /usr などの疎ルートゾーンに書き込んでいる場合や、/opt などの非デフォルトファイルシステムが疎ルートゾーンで共有されている場合には、コンポーネントの設定が失敗することがあります。

注 – ほとんどすべての Java ES 製品コンポーネントは、デフォルトでは疎ルートゾーン内で書き込み可能である、/opt の下にインストールされます。詳細は、『Sun Java Enterprise System 5 インストールリファレンス (UNIX 版)』を参照してください。

現時点で、おおよそ 20 個の Java ES 製品コンポーネントについて、大域ゾーンと非大域ゾーンとの間でライフサイクル管理と設定/ランタイム管理の分離をサポートできるようになっていません。各種製品コンポーネントで設定とアップグレードに異なる方法が採用されています。このような事情により、Message Queue 以外の Java ES 製品コンポーネントの伝播は、現時点ではサポートされていません。詳細は、52 ページの「Java ES 伝播ポリシー」を参照してください。

Java ES インストーラでのゾーンサポート

48 ページの「Java ES にゾーンを使用する理由」で論じた使用のシナリオ、および49 ページの「Java ES コンポーネントのゾーン制限」で論じた Java ES コンポーネントの要件と制限に基づいて、Java ES インストーラは、Java ES 製品コンポーネントのインストールおよびアップグレードと共有コンポーネントの同期に適したゾーンサポートを提供します。問題となりうるインストールおよびアップグレードシナリオを回避するために、インストーラにはポリシーも実装されています。

Java ES 伝播ポリシー

セクション 3 で論じた制限に基づいて、Java ES インストーラは次の 2 つの Java ES 伝播ポリシーを実装しています。

- 製品コンポーネントが大域ゾーンにインストールされたなら、それらは非大域ゾーンへと伝播しないようにデフォルトで設定されます (Message Queue は例外)。そのため、それらのコンポーネントは非大域ゾーンのレジストリに表示されず、インストールされたコンポーネントにアクセスすることもできません。
- 製品コンポーネントのインストールなどの一環として共有コンポーネントが大域ゾーンにインストールされると、共有コンポーネントは非大域ゾーンに伝播するよう設定されます。それにより、それらのコンポーネントは非大域ゾーンのレジストリに表示され、インストールされた共有コンポーネントにアクセスできるようになります。このポリシーは、49 ページの「Java ES 共有コンポーネントとゾーン」で説明されているように、ゾーン内では共有コンポーネントのバージョンを同期させなければならないという要件を実施するのに役立ちます。

製品コンポーネントのインストール

Java ES インストーラでは、製品コンポーネントとともに、各製品コンポーネントをサポートするために必要な共有コンポーネントもインストールできます。選択した製品コンポーネントをインストールする前に、インストーラは現行バージョンと以前のバージョンの共有コンポーネントがあるかどうかを確認します。インストーラが選択したコンポーネントに必要な共有コンポーネントが古いバージョンであるか存在しないことを検出すると、インストーラは現在インストールされているすべての共有コンポーネントをアップグレードし、選択したコンポーネントに必要で存在していない共有コンポーネントをすべてインストールします。この動作は50 ページの「共有コンポーネントの同期」の要件を満たすものであり、非ゾーンオペレーティングシステム、大域ゾーン、およびすべての非大域ゾーンに適用されます。

しかし、この動作には次の 2 つの例外があります。

- 疎ルートゾーンでは、一部の共有コンポーネントをインストールまたはアップグレードできないため (51 ページの「共有コンポーネントと疎ルートゾーン」を参照)、それらの共有コンポーネントを大域ゾーンでインストールまたはアップグレードするまでインストールは停止します。インストーラは次のようなメッセージを表示します。「選択したコンポーネントに必要となる次の共有コンポーネントは、疎ルートゾーンではインストールまたはアップグレードできません。それらの共有コンポーネントを大域ゾーンでインストールまたはアップグレードしてから、続行してください。「すべての共有コンポーネント」オプションを使用してください。」詳細は、55 ページの「すべての共有コンポーネント」を参照してください。
- 大域ゾーンでは、非大域ゾーンが存在する場合、現在インストールされているすべての共有コンポーネントをアップグレードして、選択したコンポーネントに必要で存在していない共有コンポーネントをインストールすることは行われず、インストーラは特定の製品コンポーネントにとって必要かどうかにかかわりなく、すべての Java ES 共有コンポーネントを同期します。こうすることで、すべての共有コンポーネントは非大域ゾーンに伝播されるので、非大域ゾーンでバージョンの異なる共有コンポーネントが混在することはありません。

製品コンポーネントのアップグレード

Java ES Release 5 には、製品コンポーネントをアップグレードする新しい機能が実装されており、Application Server、Message Queue、HADB、および Java DB に限って有効です。Java ES インストーラは以前にインストールされたこれらの製品コンポーネントのリリースバージョンを検出し、それらがアップグレード可能であることを「コンポーネントの選択」ページに示します。これら 4 つの製品コンポーネントから選択すると、インストーラは新規インストールに使用するのと同種の論理を使用して、それらのコンポーネントをアップグレードします。

詳しく説明すると、選択した製品コンポーネントをアップグレードする前に、インストーラは現行バージョンと以前のバージョンの共有コンポーネントがあるかどうかを確認します。インストーラが選択したコンポーネントに必要な共有コンポーネントが古いバージョンであるか存在しないことを検出すると、インストーラは現在インストールされているすべての共有コンポーネントをアップグレードし、選択したコンポーネントに必要で存在していない共有コンポーネントをすべてインストールします。この動作は55 ページの「すべての共有コンポーネント」で説明する要件を満たすものであり、非ゾーンオペレーティングシステム、大域ゾーン、およびすべての非大域ゾーンに適用されます。

しかし、この動作には次の 3 つの例外があります。

- 疎ルートゾーンでは、一部の共有コンポーネントをインストールまたはアップグレードできないため、それらの共有コンポーネントを大域ゾーンでインストールまたはアップグレードするまでアップグレード操作は停止します (詳細は、51 ページの「共有コンポーネントと疎ルートゾーン」を参照)。インストーラは次のようなメッセージを表示します。「選択したコンポーネントに必要となる次の共有コンポーネントは、疎ルートゾーンではインストールまたはアップグレードできません。それらの共有コンポーネントを大域ゾーンでインストールまたはアップグレードしてから、続行してください。「すべての共有コンポーネント」オプションを使用してください。」(詳細は、55 ページの「すべての共有コンポーネント」を参照)。
- Application Server と Message Queue はともに Solaris オペレーティングシステムにバンドルされています。これらのいずれのバージョンも、疎ルートゾーンでは直接アップグレードできません。これらのバンドルされている 2 つのコンポーネントの詳細については、61 ページの「製品コンポーネントの特殊ケース」を参照してください。
- 大域ゾーンでは、非大域ゾーンが存在する場合、現在インストールされているすべての共有コンポーネントのアップグレードと、インストールを選択したコンポーネントに必要で存在していない共有コンポーネントのインストールは行われず、その時点でインストールを選択したコンポーネントに必要かどうかにかかわらず、インストーラがすべての Java ES 共有コンポーネントを同期します。こうすることで、すべての共有コンポーネントは非大域ゾーンに伝播されるので、非大域ゾーンでバージョンの異なる共有コンポーネントが混在することはありません。

注 – 非大域ゾーンの製品コンポーネントのインストールやアップグレードを妨げる、数多くの特殊なケースや例外があります。このようなケースについては、61 ページの「特殊なケースまたは例外」で説明しています。

すべての共有コンポーネント

すべての共有コンポーネントを同期させなければならない状況に対応した、すべての共有コンポーネントの同期オプションが提供されています。「すべての共有コンポーネント」オプションを選択すると、インストーラは特定の製品コンポーネントにとって必要かどうかにかかわりなく、現在インストールされているすべての共有コンポーネントをアップグレードし、存在していない共有コンポーネントがあればインストールします。このオプションは大域ゾーンと完全ルートゾーンには適用されますが、疎ルートゾーンには適用されません。

「すべての共有コンポーネント」オプションは、次のようなゾーンに基づいた 2 つのシナリオで必要になります。

- 製品コンポーネントの手動によるアップグレード「すべての共有コンポーネント」オプションは、Java ES インストーラではアップグレードできない製品コンポーネントをアップグレードするときに必要となる、共有コンポーネントのインストールとアップグレードを行うために必要です。
- 疎ルートゾーンでのインストールまたはアップグレード一部の共有コンポーネントは、デフォルトの疎ルートゾーンにインストールできません (詳細は、53 ページの「製品コンポーネントのインストール」および53 ページの「製品コンポーネントのアップグレード」を参照)。そのため、疎ルートゾーンでインストーラを実行する場合、関係する共有コンポーネントによっては、まず大域ゾーンで共有コンポーネントを同期することが必要になることがあります。このケースで必要な共有コンポーネントのインストールとアップグレードを実行するには、大域ゾーンで「すべての共有コンポーネント」オプションを使用します。

共有コンポーネントに関連する Java ES インストーラ動作の要約

前述の動作を要約すると、次の表のようになります。この表には、共有コンポーネントに対する Java ES インストーラの処理が、ゾーンコンテキストまたはコンポーネント選択ページでの選択にどのように左右されるかを示します。

表 A–1 共有コンポーネントに関連するインストーラ動作

ゾーンコンテキスト	製品コンポーネントを選択した場合	すべての共有コンポーネントを選択した場合
非ゾーンオペレーティングシステム	現在インストールされているすべての共有コンポーネントをアップグレードする	現在インストールされているすべての共有コンポーネントをアップグレードする
	選択した製品コンポーネントに必要で、存在していない共用コンポーネントをすべてインストールする	特定の製品コンポーネントにとって必要かどうかにかかわりなく、存在しないすべての共有コンポーネントをインストールする
大域ゾーン: 非大域ゾーンなし	現在インストールされているすべての共有コンポーネントをアップグレードする	現在インストールされているすべての共有コンポーネントをアップグレードする
	選択した製品コンポーネントに必要で、存在していない共用コンポーネントをすべてインストールする	特定の製品コンポーネントにとって必要かどうかにかかわりなく、存在しないすべての共有コンポーネントをインストールする
大域ゾーン: 非大域ゾーンあり	現在インストールされているすべての共有コンポーネントをアップグレードする	現在インストールされているすべての共有コンポーネントをアップグレードする
	特定の製品コンポーネントにとって必要かどうかにかかわりなく、存在しないすべての共有コンポーネントをインストールする	特定の製品コンポーネントにとって必要かどうかにかかわりなく、存在しないすべての共有コンポーネントをインストールする
完全ルートゾーン	現在インストールされているすべての共有コンポーネントをアップグレードする	現在インストールされているすべての共有コンポーネントをアップグレードする
	選択した製品コンポーネントに必要で、存在していない共用コンポーネントをすべてインストールする	特定の製品コンポーネントにとって必要かどうかにかかわりなく、存在しないすべての共有コンポーネントをインストールする

表 A–1 共有コンポーネントに関連するインストーラ動作 (続き)

ゾーンコンテキスト	製品コンポーネントを選択した場合	すべての共有コンポーネントを選択した場合
疎ルートゾーン	読み取り専用ディレクトリ内の一部の共有コンポーネントをアップグレードまたはインストールできない。このような共有コンポーネントを検出すると、インストーラは中断し、大域ゾーンの共有コンポーネントを管理するようユーザーに指示します。	読み取り専用ディレクトリ内の一部の共有コンポーネントをアップグレードまたはインストールできない。そのため、インストーラは中断し、大域ゾーンの共有コンポーネントを管理するようユーザーに指示します。

Java ES で推奨されるゾーンの使用法

複数ゾーン環境への Java ES の配備には製品コンポーネントが実行時に独立性を持ちリソースを効率的に活用できるようにするという大まかな目的がありますが、複数ゾーン環境を使用するより具体的な目的がいくつも存在します。これについては 48 ページの「Java ES にゾーンを使用する理由」で論じています。複数ゾーン環境での Java ES のインストールと管理の方策は、そのうちのどの目的を実現しようとしているかによって大きく左右されます。

表 A–2 では、5 つのシナリオ、それぞれに対応するインストールと管理の方策、およびそれによって実現しようとする目的を示します。これらのシナリオを組み合わせることが可能な場合もありますが、問題のある結果になったり、管理が混乱する恐れもあります。そのため、Java ES Release 5 では一般にこれらのシナリオを組み合わせた配備はサポートしていません。

また、シナリオ 1 とシナリオ 5 には問題があるため、現在 Java ES Release 5 ではサポートしていません。ただし、シナリオ 5 の場合は特定の製品コンポーネント用に適合させることが可能です。

表 A–2　Java ES のゾーンインストールおよび管理の方策

シナリオ (インストールの方策)	管理の方策	目的 (48 ページの「Java ES にゾーンを使用する理由」を参照)	コメント
1: 製品コンポーネントと共有コンポーネントを大域ゾーンにインストールし、伝播を有効にする。非大域ゾーンにはコンポーネントをインストールしない。*	コンポーネントのライフサイクル管理: 大域管理者 設定およびランタイム管理: ゾーン管理者	製品コンポーネントのライフサイクル管理を集約化 製品コンポーネントの設定およびランタイム管理を組織別に独立化	問題あり。Message Queue 以外の Java ES 製品コンポーネントにまだサポートされていない。 製品コンポーネントが、大域ゾーンへのインストールと非大域ゾーンでの設定およびランタイム管理をサポートしている必要がある。
2: 共有コンポーネントを大域ゾーンに、製品コンポーネントを完全ルートゾーンにインストールする	共有コンポーネントのライフサイクル管理: 大域管理者 製品コンポーネントのライフサイクル管理: ゾーン管理者 設定およびランタイム管理: ゾーン管理者	共有コンポーネントのライフサイクル管理を集約化 製品コンポーネントのライフサイクル、設定、およびランタイム管理の組織別の独立化	すべてのコンポーネントが同じ Java ES バージョンを実行しているか、すべての完全ルートゾーンのすべての製品コンポーネントをアップグレードする場合は、おおかた適用可能。
3: 共有コンポーネントを大域ゾーンに、製品コンポーネントを疎ルートゾーンにインストールする**	シナリオ 2 と同じ	共有コンポーネントのライフサイクルの管理を集約化 製品コンポーネントのライフサイクル、設定、およびランタイム管理の組織別の独立化 シナリオ 2 のリソース効率をさらに向上 (47 ページの「完全ルートゾーンと疎ルートゾーンのどちらを選択するか」を参照)	疎ルートゾーンに製品コンポーネントをインストールする場合に、このシナリオを推奨します。一部の共有コンポーネントは疎ルートゾーンにインストールできない場合があるため、大域ゾーンへのインストールが必要となります。
4: 製品コンポーネントと共有コンポーネントを完全ルートゾーンにインストールする	コンポーネントのライフサイクル管理: ゾーン管理者。設定およびランタイム管理: ゾーン管理者	バージョンの分離	どの共有コンポーネントおよび製品コンポーネントも大域ゾーンにインストールしないようにします。完全ルートゾーンの場合に推奨されるシナリオです。

表 A–2　Java ES のゾーンインストールおよび管理の方策　(続き)

シナリオ (インストールの方策)	管理の方策	目的 (48 ページの「Java ES にゾーンを使用する理由」を参照)	コメント
5: 製品コンポーネントと共有コンポーネントを疎ルートゾーンにインストールする	シナリオ 4 と同じ	製品コンポーネントのライフサイクル、設定、およびランタイム管理の組織別の独立化 シナリオ 4 のリソース効率をさらに向上 (47 ページの「完全ルートゾーンと疎ルートゾーンのどちらを選択するか」を参照)	問題あり。疎ルートゾーンにインストールできない共有コンポーネントが多数あるため、通常は実現不可能。

* シナリオ 1 では、ゾーン環境を完全ルートと疎ルートとに区別しません。このシナリオは、非大域ゾーンにインストールする製品コンポーネントがないことを前提としています。非大域ゾーンへの製品コンポーネントのインストールは、シナリオ 2 ～ 5 で扱います。

** シナリオ 3 は、`/opt` が疎ルートゾーン内の読み取り専用ディレクトリになっていないことを前提とします。`/opt` が読み取り専用であると、ほとんどの Java ES 製品コンポーネントは疎ルートゾーンにインストールできないため、代わりにシナリオ 1 のように、それらを大域ゾーンにインストールすることが必要になります。

推奨されている実践事項

表 A–2 を踏まえ、推奨されている実践事項を次にいくつか紹介します。

- 48 ページの「Java ES にゾーンを使用する理由」にある目的の中から達成しようと思うものを選び、それを基にして Java ES ゾーンの配備方針を前もって計画します。表 A–2 のさまざまなシナリオに示されているように、目的が違えば、インストールや管理の方針も違ってきます。
- シナリオを組み合わせることは避けてください。特に以下のオプションがあります。
  - Java ES ゾーンの配備と管理の方針をできるだけシンプルに保ちます。Java ES コンポーネントの完全ルート配備と疎ルート配備を同じコンピュータに混在させないでください。シナリオ 3 のような疎ルートゾーン配備をサポートするために必要となる手順と実践事項は、シナリオ 4 のような完全ルートゾーン配備と相容れないことがあります。

- 同じ Java ES 製品コンポーネントは、バージョンが違うとしても、大域ゾーンと非大域ゾーンの両方にはインストールしないでください。シナリオ 1 のような大域ゾーンのインストールシステムをアップグレードするために必要な手順は、シナリオ 4 のような非大域ゾーンのインストールシステムを破壊する場合があります。
- Release 4 以前の Java ES コンポーネントが完全ルートゾーンにインストールされている場合は、Java ES Release 5 コンポーネントは製品コンポーネントも共有コンポーネントも大域ゾーンにインストールせず、大域ゾーンの Java ES コンポーネントを Release 5 にアップグレードしないでください。言い換えると、完全ルートゾーンに Java ES インストールシステムがすでに存在しているなら、シナリオ 2 はサポートされません。大域ゾーンでインストールまたはアップグレードを行うと、完全ルートゾーンに Release 4 と Release 5 のファイルが混在してしまう可能性があります。

- インストールで推奨されている実践事項:
  - 複数のゾーンでそれぞれ異なる Java ES 製品コンポーネントを実行する場合は、製品コンポーネントを非大域ゾーンにインストールします (シナリオ 2、3、4、5)。
  - 複数のゾーンでそれぞれ異なる Java ES 製品コンポーネントを実行するものの、大域ゾーンで共有コンポーネントのライフサイクルの集中管理および共有コンポーネントの同期を行う場合は、製品コンポーネントを非大域ゾーンにインストールします (シナリオ 2、3)。製品コンポーネントを疎ルートゾーンにインストールする場合には必ず実践するよう推奨されています。
  - Java ES 製品コンポーネントをバージョンごとに分離させる場合、または他の理由で Java ES 製品コンポーネントの配備をそれぞれ独立させる場合は (シナリオ 4)、すべての Java ES コンポーネントを完全ルートゾーンにインストールし設定します。どの Java ES コンポーネントも大域ゾーンにインストールしないでください。
- アップグレードで推奨されている実践事項:
  - インストールされているすべての Release 4 製品コンポーネントを Release 5 にアップグレードする場合、大域ゾーンのすべての Java ES 共有コンポーネントを同期してから、インストールされているゾーンにある希望する製品コンポーネントのアップグレードを実行します。Release 5 の共有コンポーネントには下位互換性があります。
  - Release 4 または Release 5 製品コンポーネントを非大域ゾーン環境にインストールしてあり、その環境に非大域ゾーンを追加し、その新しい非大域ゾーンに製品コンポーネントをインストールしようと思う場合は、必ず前述の推奨されている実践事項に沿った仕方で行ってください。したがって、場合によっては、大域ゾーンのコンポーネントをアンインストールして非大域ゾーンにインストールすることも必要になります。

配備アーキテクチャー

表 A–2 のシナリオの説明と前述の推奨されている実践事項には、複数ゾーン環境用に推奨されている Java ES 配備アーキテクチャーに関する内容は含まれていません。このアーキテクチャーは、複数台のコンピュータが接続されているネットワーク環境用に作成された配備アーキテクチャーの改良版です。別の言い方をすれば、複数ゾーン環境を利用するからといって、Java ES 配備システムの高パフォーマンス、高可用性、スケーラビリティー、セキュリティー、および保守性を実現する基本的な配備設計の手法は変わりません。複数ゾーン環境を使用することで、このような配備アーキテクチャーを整理統合してより少ない台数のコンピュータで運用することが可能になります。

しかし、これまでのセクションで論じられてきたように、具体的にどのように Java ES 配備アーキテクチャーを複数ゾーン環境に改良するかは、採用する管理方針によって大きく左右されます。また、配備アーキテクチャーは高可用性を実現する方策にも依存します。

表 A–2 と前述の推奨されている実践事項には、説明されているシナリオを実装する推奨手順は含まれていません。場合によっては、Java ES コンポーネントをインストールする順序と非ローカルゾーンを作成する順序が重要になることがあります。

特殊なケースまたは例外

一部の Java ES 共有コンポーネントと一部の Java ES 製品コンポーネントが Solaris 10 にバンドルされていることが主な原因となっている特殊なケースがいくつかあります。このバンドルが理由となって、これらの Java ES コンポーネントは大域ゾーン、さらには大域ゾーンから作成される任意の非大域ゾーンに存在します。

製品コンポーネントの特殊ケース

- Message Queue は Solaris 10 にバンドルされているため、先に大域ゾーンから Message Queue を削除していなければ、非大域ゾーンが作成されるときに Message Queue は自動的に伝播されます。Message Queue は疎ルートゾーンにインストールできません。Message Queue は、デフォルトで Java ES インストーラによって大域ゾーンでインストールまたはアップグレードされると、他の製品コンポーネントとはことなり、非大域ゾーンに伝播されます。
- Application Server は Solaris 10 にバンドルされているため、先に大域ゾーンから Application Server を削除していなければ、非大域ゾーンが作成されるときに Application Server は自動的に伝播されます。このようにして伝播されると、バンドルされている Application Server (`/usr` にインストールされている) は、`/usr` がデフォルトで読み取り専用である疎ルートゾーンでは、Java ES インストーラを使用してアップグレードできません。この問題に対処するため、疎ルートゾーンに

Release 5 Application Server をインストールする前に、バンドルされている Application Server を手動で大域ゾーンから削除する必要があります。

- Sun Cluster は大域ゾーンにのみインストールできます。Sun Cluster は非大域ゾーンではサポートされていません。

共有コンポーネントの特殊ケース

- Solaris 10 (Update1 および Update 2) にバンドルされている SJWC パッケージは Java ES インストーラでは削除できません。これらの古い SJWC パッケージでは `SUNW_PKG_ALLZONES` が True に設定されており、これはパッケージがすべてのゾーンで同一でなければならず、大域管理者によってのみ管理可能であることを意味します。したがって、大域ゾーンにあるこれらのパッケージは手動で削除して正しいパッケージで置き換える必要があります。

  Java ES インストーラが非大域ゾーンに選択したコンポーネントをインストールしようとしているところで、SJWC のアップグレードが必要であることを検出すると、インストーラは停止します。この問題は、Solaris 10 の Update 1 および 2 へのインストール時に発生します。

  回避方法として、古い SJWC パッケージを大域ゾーンから削除して、ゾーン伝播属性が正しく設定されている SJWC 2.2.6 に置き換える特別なスクリプトが開発されています。結果として、SJWC 2 2.6 はすべての非大域ゾーンに伝播されます。
- 共通エージェントコンテナ。バージョン 1.1 は、Sun Cluster、Sun Cluster GE、または Sun Cluster Agents がインストールされている場合にのみインストールされます。「すべての共有コンポーネント」オプションが選択されている場合は、インストールされません。その場合はバージョン 2.0 のみがインストールされます。
- Sun Explorer データコレクタ。この共有コンポーネントは、Sun Cluster、Sun Cluster GE、または Sun Cluster Agents がインストールされている場合にのみインストールされます。「すべての共有コンポーネント」オプションが選択されている場合は、インストールされません。

サンプル: 疎ルートゾーンに Application Server をインストールする

次の例は、Java ES ゾーンサポートに伴う複雑さの一面を描き出すために用意されています。この例では、目的は Solaris 10 疎ルートゾーンに Application Server をインストールすることです。このインストールは、Application Server と、Application Server が依存関係にある Message Queue が Solaris 10 にバンドルされており複雑であるため、バンドルされているバージョンはすべての非大域ゾーンにインストールされます。詳細は、61 ページの「製品コンポーネントの特殊ケース」を参照してください。

Application Server を疎ルートゾーンにインストールするには、まずバンドルされているバージョンを削除する必要があります。疎ルートゾーンのバンドルされている

バージョンは、読み取り専用ディレクトリにインストールされているため、アップグレードが単純ではありません。バンドルされているバージョンを疎ルートゾーンから削除するには、それを大域ゾーンで削除する必要があります。

さらに、Message Queue は大域ゾーンにインストールされているので、表 A–2 のシナリオ 3 から出発することになり、その場合製品コンポーネントはインストールされず、共有コンポーネントのみが大域ゾーンにインストールされます。ただし、Message Queue は読み取り専用ディレクトリにインストールするため、疎ルートゾーンにインストールすることはできず、大域ゾーンでインストールとアップグレードを行う必要があります。

手順は次のとおりです。

1. ご使用のシステム上で Solaris 10 が実行中であることを確認します。

   この例では、大域ゾーンに Java ES コンポーネントが何も明示的にインストールされていない、クリーンバージョンの Solaris 10 を想定しています。

2. 疎ルートゾーンを作成し、設定、インストール、およびブートを行います。

   このゾーンにはすでに大域ゾーンにインストールされている Java ES コンポーネント、つまり Solaris 10 にバンドルされているバージョンの Message Queue と Application Server がすべて組み込まれます。

3. バンドルされているバージョンの Application Server を大域ゾーンから削除します。

   このことは、次のコマンドを使用して Application Server パッケージを手動で削除することにより行う必要があります。

   ```
   pkgrm SUNWascmnse SUNWaslb SUNWasut ...
   ```

   ここで、次のコマンドを使用することにより、完全セットのパッケージを入手できます。

   ```
   pkginfo -I|grep -I application server
   ```

   結果として、次のパッケージが組み込まれます。

   ```
   SUNWascmnse, SUNWaslb, SUNWasut, SUNWasac, SUNWasdem, SUNWasman, SUNWaswbcr,
   SUNWasacee, SUNWashdm, SUNWasmanee, SUNWascml, SUNWasJdbcDrivers, SUNWasu,
   SUNWascmn, SUNWasjdoc, SUNWasuee
   ```

   また、次のローカリゼーションパッケージも組み込まれる場合があります。

   ```
   SUNWLocaleasacee, SUNWLocaleascmnse, SUNWLocaleasu, SUNWLocaleasuee
   ```

   大域ゾーンからの Application Server の削除がステップ 2 で作成された疎ルートゾーンに伝播します。このステップとステップ 2 の順序を入れ替えることも可能です。

4. Java ES 5 共有コンポーネントを大域ゾーンにインストールします。

   a. Java ES インストーラを大域ゾーンで実行します。

b. コンポーネント選択パネルから「すべての共有コンポーネント」を選択します。それ以外のコンポーネントは選択しないでください。

c. 共有コンポーネントの同期を完了します。これで、すべての共有コンポーネントが大域ゾーンに同期され、すべての非大域ゾーンに伝播されました。

5. 大域ゾーンの Message Queue をアップグレードします。

   Solaris 10 にバンドルされているバージョンの Message Queue は、ステップ 2 によってすでに疎ルートゾーンにインストールされています。疎ルートゾーンの Message Queue をアップグレードするには、大域ゾーンの Message Queue をアップグレードするだけでよく、そのアップグレードが疎ルートゾーンに伝播します。Message Queue は、疎ルートゾーンにインストールできないものの、大域ゾーンにインストールすると非大域ゾーンに伝播する唯一の製品コンポーネントです。

   a. Java ES インストーラを大域ゾーンで実行します。

   b. コンポーネント選択パネルで Message Queue を選択します。それ以外のコンポーネントは選択しないでください。

   c. Message Queue のアップグレードを完了します。

6. Application Server を疎ルートゾーンにインストールします。

   a. Java ES インストーラを疎ルートゾーンで実行します。

   b. コンポーネント選択パネルで Application Server を選択します。アップグレードにそれ以外のコンポーネントを選択しないでください。Message Queue が選択されている場合は、それを解除してください。

   c. Application Server のインストールを完了します。

索引

A

Access Manager
  LDAP スキーマの変更，21
  依存性のリスト，28
  サードパーティー製 Web コンテナ，39
Access Manager SDK, 依存性のリスト，29
Apache Web Server，39
Application Server, 依存性のリスト，29

B

BEA WebLogic，39
BEA WebLogic Server
  Portal Server の依存性，30

C

Calendar Server, LDAP スキーマの拡張，21
CPU 要件，17

D

Directory Preparation Tool
  LDAP スキーマの拡張，21, 35
Directory Proxy Server, 依存性のリスト，30
Directory Server
  Access Manager によって変更されるスキーマ，21, 34
  依存性のリスト，30
  デフォルト LDAP スキーマ，20-21, 34
Directory Server *(続き)*
  マルチマスターレプリケーション，34

H

HADB，39
High Availability Session Store (HADB), Application Server のローカル依存性，30

I

IBM WebSphere，39
IBM WebSphere Application Server
  Portal Server の依存性，30

L

LDAP スキーマ
  Access Manager による変更，34
  Delegated Administrator による拡張，21
  Directory Preparation Tool による拡張，21, 35
  Java ES インストールによる確立，20, 34
  デフォルト，20-21
LDAP ディレクトリ
  Directory Server による提供，21
  ディレクトリツリーの確立，21
  ほかのディレクトリソフトウェアによる提供，23
LDAP ディレクトリツリー
  Java ES インストールによる確立，20

LDAP ディレクトリツリー *(続き)*
インストーラによって確立されるベースサフィックス，22

M
Message Queue, Application Server のローカル依存性，29
Messaging Server, LDAP スキーマの拡張，21

P
Portal Server, 依存性のリスト，30
Portal Server Secure Remote Access, 依存性のリスト，31

R
RAM 要件，17

S
Sun Cluster Agents, 依存性のリスト，31
Sun Cluster ソフトウェア, インストール手順，33

W
Web Proxy Server, 依存性のリスト，31
Web Server, Application Server のローカル依存性，29
Web コンテナ, 依存性，27

い
依存性
Web コンテナ，27
依存性の理由，26
インストール計画の順序の決定，27
ソリューションレベルとローカル，27
依存性 *(続き)*
表，32
インストーラ, 分散インストールのための使用方法，26
インストール, 上位レベルのタスク，13
インストール計画
コンポーネントの依存性によって決定されるシーケンス，27
コンポーネントの相互動作，32
作成方法，40
ニーズ，25, 40
分散インストール向け，26
例，42
インストールのタスク，13

こ
コンポーネントの相互動作
インストール計画による実現，25
設定，32

さ
サードパーティー製 Web コンテナ，39
サービス品質要件
冗長化による達成，33
冗長性による達成，16
達成するためのハードウェアの選択，17
例，17

し
仕様
オペレーティングシステム，17
コンピュータハードウェア，17
ネットワーク接続，18
シングルユーザーエントリ，39

す
スキーマ 1，39

て
ディスク要件，17

ね
ネットワーク接続仕様, 例，18

は
配備アーキテクチャー
　分析，15-17
　例，15

ひ
非ルートインストール，40

ふ
負荷分散, インストール手順，33

へ
ベースサフィックス, インストーラによる確立，22

ま
マルチマスターレプリケーション, インストール
　手順，34

よ
用語集、リンク，7